

A MODEL VALIDATION METHODOLOGY FOR ISOLATING INCONSISTENT
KNOWLEDGE BETWEEN FUZZY RULE-BASED AND QUANTITATIVE
MODELS USING FUZZY SIMULATION

By

GYOONSEOK KIM

A DISSERTATION PRESENTED TO THE GRADUATE SCHOOL
OF THE UNIVERSITY OF FLORIDA IN PARTIAL FULFILLMENT
OF THE REQUIREMENTS FOR THE DEGREE OF
DOCTOR OF PHILOSOPHY

UNIVERSITY OF FLORIDA

1998

*To my parents,
my wife, Jungsook, my daughter, Karyoon and my son, Seungyup*

ACKNOWLEDGEMENTS

I wish to express my gratitude to my supervisor and committee chairman, Dr. P. A. Fishwick, for his invaluable guidance, encouragement and patience throughout the past four years of my graduate study. I would like to express my sincere appreciation to Dr. Douglas D. Dankel, Dr. L. M. Fu, Dr. S. Rajasekaran, and Dr. S. X. Bai for serving on my supervisory committee and providing invaluable suggestions for my research.

I am grateful to all my colleagues at the Department of Computer and Information Science and Engineering for their help and friendship. I thank Dr. Jinjoo Lee who helped me a lot when I first joined the research group. Special thanks go to Youngsup Kim and Kangsun Lee in my group for their suggestions and companionship during my research. I wish all of them the best of luck in their future endeavors.

Last but most importantly, I am deeply indebted to my parents, my wife Jungsook, my daughter Karyoon and my son Seungyup for their encouragement, love and understanding. My greatest gratitude is to them.

TABLE OF CONTENTS

ACKNOWLEDGEMENTS	iii
ABSTRACT	vi
CHAPTERS	1
1 INTRODUCTION	1
1.1 Problem Statement	2
1.2 Purpose of Research	5
1.3 Contribution to Knowledge	7
1.4 Outline	7
2 BACKGROUND AND RELATED WORK	9
2.1 Modeling Process in Simulation and Fuzzy Set Literature	9
2.1.1 Simulation Modeling Process	9
2.1.2 Fuzzy Modeling Process	12
2.2 Fuzzy Set Theory and Its Application	14
2.2.1 Notation, Terminology, and Basic Operations	15
2.2.2 Membership Function Construction	19
2.2.3 Fuzzy Controller	21
2.3 Fuzzy Set Theory in Computer Simulation	27
3 A NEW FUZZY SIMULATION APPROACH	33
3.1 Expert Rule Format	33
3.2 Fuzzy Simulation	34
3.2.1 Fuzzy Simulation for Simplex Rules	35
3.2.2 Fuzzy Simulation for Compound Rules with Arithmetic Operations	39
3.2.3 Fuzzy Simulation for Compound Rules with Logic Operations	42
4 A METHOD FOR ISOLATING INCONSISTENCY	46
4.1 Measurements of Inconsistency	48
4.2 Checking Consistency When MF_{expert} is Available	49
4.3 Checking Consistency When MF_{expert} is Unavailable	50
4.3.1 Process in General	50
4.3.2 Heuristic Function and Search Method for Generating Approximate MF_{fuzzy}	51
4.3.3 Various Forms of Expert's Estimates on Linguistic Terms	54

4.3.4	Algorithm to generate MF_{fuzzy}	57
4.3.5	Identify Inconsistent Rules	61
4.4	Time Complexity	64
4.4.1	Time Complexity for Fuzzy Simulation	64
4.4.2	Time Complexity for MF_{fuzzy} Generation	65
5	FULTON: STEAMSHIP MODELING	67
5.1	Quantitative Model of Boiler Assembly	67
5.2	Qualitative Model of Boiler Assembly	69
5.3	Checking Consistency When MF_{expert} is Available	70
5.4	Checking Consistency When MF_{expert} is Unavailable	74
5.4.1	The Case Where Approximate MF_{fuzzy} is Successfully Generated	74
5.4.2	The Case Where MF_{fuzzy} is Unsuccessfully Generated	80
5.4.3	Human Intervention	81
6	PREDATOR-PREY POPULATION	86
6.1	Qualitative Model	86
6.2	Quantitative Model	87
6.3	Consistency Checking	91
6.4	MF_{fuzzy} Generation	95
7	FUTURE WORK	99
7.1	Limitations and Improvements	99
7.1.1	Resolving Inconsistency	99
7.1.2	Performance Index	99
7.1.3	Local Optimality During MF_{fuzzy} Generation	100
7.2	Application	101
7.2.1	Application in Control Industries	101
7.2.2	Application in MOOSE	102
8	CONCLUSION	106
	REFERENCES	108
	BIOGRAPHICAL SKETCH	112

Abstract of Dissertation Presented to the Graduate School
of the University of Florida in Partial Fulfillment of the
Requirements for the Degree of Doctor of Philosophy

A MODEL VALIDATION METHODOLOGY FOR ISOLATING INCONSISTENT
KNOWLEDGE BETWEEN FUZZY RULE-BASED AND QUANTITATIVE
MODELS USING FUZZY SIMULATION

By

GYOONSEOK KIM

August, 1998

Chairman: Dr. Paul A. Fishwick

Major Department: Computer and Information Science and Engineering

Model validation is a complicated and multifaceted procedure in which all possible information such as real-world data, other model input-output data, compiled knowledge in the form of mathematical models and expert opinions can be *made consistent*. Even though validation using real-world data provides the ideal case, in reality, obtaining such data is not always possible. In this dissertation, we assume that expert opinion is presented in the form of fuzzy rules. The fuzzy simulation approach presented here provides a mechanism for directly encoding uncertainty from human reasoning into a computer simulation by mapping the fuzzy linguistic values of the expert's rules into simulation components. To perform a quantitative comparison between the two kinds of models, quantitative measures have been formulated to gauge the sources and the degree of inconsistency. Using the fuzzy simulation approach and the quantitative measures, this method provides an interactive environment for *isolating inconsistency* between the fuzzy rule-based model and the

quantitative model allowing some degree of consistency. This environment also facilitates *humans in the loop* resolving inconsistency by helping them to identify and revise the most inconsistent component rapidly and then analyze the effectiveness of that modification. Through checking consistency between these two knowledge representations, our approach serves as a method of *checks and balances* to enhance the complex model validation process.

CHAPTER 1 INTRODUCTION

While model verification deals with building the model *right*, model validation deals with building the *right* model [1]. Validation is concerned with determining whether the model is an accurate representation of the system under study [29]. Model validation is part of the total model development process, and it consists of performing a series of tests and evaluations within the model development process. This validation process is multifaceted and involves the minimal procedure of taking a set of real-system observations and rectifying these observations with an assumed mathematical model or vice versa. This process involves an estimation of parameters of the model that yields a model that best reflects the real-system behavior.

In practice, validation is a complicated process, since there may be numerous sources of knowledge comparable with the model output that is under investigation. For many fields, most notably medicine, sociology and economics, there may be a plethora of knowledge and data. This knowledge may be quantitative or qualitative, or both, with experts providing opinions that must somehow be reconciled with quantitative proposed models known to characterize a similar real-world problem. However, currently no algorithm or procedure is available to identify suitable validation techniques [45]. Moreover, one does not always have a complete set of data and a set of models waiting to be identified. The data may be just as incomplete as the model suppositions. The difficulty of achieving the data validity for model validation is discussed in [29].

We have created a methodology that enhances the validation process for such situations. In particular, our method assumes that there is at least one fuzzy rule-based

model and one proposed quantitative model. The rule-based model is composed of a set of *N* IF-THEN statements, and the quantitative model is defined as a model whose parameters and state variables take real values. Our method locates inconsistencies between these two representations, and we created an interactive tool for partially resolving inconsistencies. Comparing and contrasting the expert rules with the quantitative model is viewed as being an integral part of an ongoing system validation procedure.

1.1 Problem Statement

Knowledge about a given physical system is often obtained from experts in the form of rules. Although the rule-based model is occasionally associative or shallow in nature, this model can easily capture human heuristic and problem-solving knowledge in an efficient way [21, 5, 48]. Fuzzy set theory [57, 58, 59, 27, 14, 61] provides linguistic IF-THEN rules to make use of such experts' knowledge and experience naturally. Through a suitable fuzzy inference scheme, the rule-based system provides a solution without exploiting underlying causal relations on which the solution is based.

In some cases, a quantitative model exists which represents all or part of the behaviors of the physical system. This model can be a mathematical model where the system behavior is characterized by one or a series of equations or inequalities. Or this model may be a simulation model where each change in the status of the system is captured over time. This kind of quantitative model provides deeper and more theoretical knowledge when expert system developers need to find solutions for technical problems [5, 16, 48].

To compare and contrast the qualitative model with the quantitative model to serve model validation, we should provide answers for the following fundamental questions: *how much do the models differ?* and *how can one address the difference*

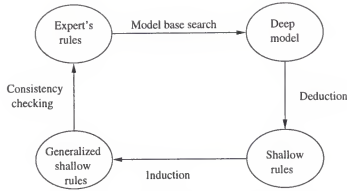


Figure 1.1. A knowledge acquisition cycle

between the two models? One way of handling the inconsistencies between the two different levels of models is to form a *knowledge acquisition cycle* as in Figure 1.1 [5]. Approaches for creating *model bases* are discussed within the context of computer simulation [15, 18]. For example, the model base represents compiled knowledge about many domains, such as a mathematical queuing model for waiting line problems. If a match is found during the model base search, then shallow rules (or input-output pairs) are generated by means of deduction process. For such a process, exhaustive simulations are executed based on this deep model. Since the size of the shallow rules in Figure 1.1 resulting from the deduction process is usually too large for a human to study and validate against the original expert's rules, fuzzy induction or fuzzy system identification methods [31, 51, 38, 50, 49] can be employed to obtain a more comprehensible and generalized set of linguistic rules. After completing these processes, the rules obtained as a final product become suitable for a human to study and compare against the expert's original rules.

However, the fundamental problems of the knowledge acquisition cycle mentioned above can be described as

1. First and most importantly, forming the above knowledge acquisition cycle to check consistency requires a series of difficult tasks as well as a long process.

2. For the direct comparison between the generalized rules in Figure 1.1 and the expert rules, the two rule sets should have the same rule structure, linguistic variables and values. This means that fuzzy induction or identification methods should take the expert's rules into account when they generate the rules. However, most approaches take numerical input and output data to generate fuzzy rules without considering the expert's prior knowledge about the system. Consequently, it is difficult to measure the inconsistency between the two models quantitatively. This could make a decision-making process quite complicated if one wants to resolve inconsistency, especially when a large portion of knowledge components shows inconsistency.
3. Identifying fuzzy rules using only numerical data without properly assumed rule-structure information easily suffers from the *curse of dimensionality* [32], in which the number of possible rules increases exponentially with the number of possible variables. Moreover, searching for influential variables among all possible variables based only on the numerical data causes the problem of combinatorial optimization.
4. For the deduction process in Figure 1.1, all fuzzy simulation approaches [4, 46, 17] require that proper linguistic definitions (i.e., fuzzy membership functions) be defined *a priori* before performing fuzzy simulation. Since such definitions of the linguistic terms are difficult to obtain even from experts because of the uncertainty arising from linguistic vagueness, searching for the proper membership functions belongs to the general problem area of knowledge acquisition within the underlying framework of fuzzy set theory [27]. Thus, if such definitions are not available *a priori*, this additional knowledge acquisition process will clearly become a *bottleneck* in the cycle shown in Figure 1.1.

1.2 Purpose of Research

This research is concerned with devising a method for isolating inconsistency between the two different levels of models in an efficient and systematic manner. To achieve this goal, we have developed a knowledge acquisition cycle as shown in Figure 1.2 [26, 24, 25]. Using this approach, we can alleviate the problems discussed in the previous section.

- First of all, the knowledge acquisition cycle presented here forms a simple process compared to the one in Figure 1.1. The *fuzzy simulation* approach introduced here directly encodes uncertainty arising from human linguistic vagueness into simulation components and utilizes *quantitative* models for the deduction process. Since this method accepts the linguistic values in the expert's rule premises as simulation inputs and produces linguistic values as outputs using the same terminologies that the expert used in his (or her) rules, the direct comparison between these two models is possible without an additional induction step.
- Next, by taking into account the expert's prior knowledge about the system, the major variables affecting the system don't need to be identified. This alleviates major problems such as high computational complexity and local optimality, which usually arise from the structure identification process.
- Finally, to alleviate the knowledge acquisition bottleneck, this method allows users to employ various levels of estimates depending on the linguistic vagueness in their rules.

As shown in Figure 1.2, this method isolates inconsistency through the two phases:

- 1) *consistency checking* and 2) *interactive user control*. In the *consistency checking*

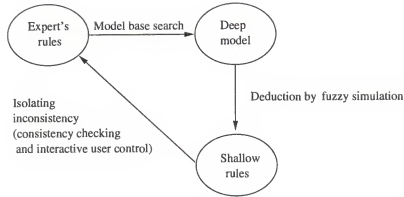


Figure 1.2. A knowledge acquisition cycle using fuzzy simulation

phase, depending on the expert's various levels of linguistic vagueness (for example, central points, intervals or fuzzy sets), the result of fuzzy simulations is directly compared against the expert's rules (when fuzzy sets are provided), or fuzzy set definitions, which properly *fill in* the expert's knowledge, are searched through incremental optimization over fuzzy space (when central points or interval estimates are provided). In both cases, the method suggests the source and the amount of inconsistency to users using quantitative measures we have formulated. If the amount of inconsistency exceeds a reasonable range, human intervention is possible via *interactive user control*: users may modify either expert rules or simulation model components to reduce the amount of inconsistency. Even at this point, the quantitative measures mentioned above help users identify and revise the most inconsistent component rapidly and analyze the effectiveness of that modification. This human-machine interaction allows the two models to gradually reach a consensus with a high resolution. O'Keefe [35] pointed out that this kind of *visual interaction* is one of the most promising validation techniques in expert systems.

1.3 Contribution to Knowledge

The primary contribution of this work is that, through checking consistency and resolving inconsistency, our approach serves as a method of *checks and balances* during the model validation phase of system analysis. During this process, we provide benefits to expert systems from simulation and benefits to simulation modeling from expert knowledge. In particular, when expert system researchers are studying the acquisition of deep knowledge from an expert or validating the expert's knowledge against quantitatively compiled knowledge, the first type of benefits can be obtained from simulation models [16, 15, 17, 54, 34]. The advantage from the reverse process is also obtained when simulation model validations are performed during the simulation modeling process with the aid of the expert knowledge [54, 6, 44, 45].

Additionally, using this method, we can contribute a second benefit from automatically generating fuzzy membership functions where expert rules and quantitative models match maximally. Most methods for constructing fuzzy membership functions [27, 42, 41, 9, 52, 2, 3, 33, 19] rely on a set of sample data (i.e., pairs of an element and its membership degree) from an expert's (or experts') opinions before applying curve-fitting or learning methods. However, gathering such information is not a straightforward matter, even by domain experts, if the system's input, process and response are too complex. If a quantitative model exists for the system, the method presented in this paper allows us to utilize natural rules from the domain experts instead of the restricted sample data to construct approximate fuzzy membership functions.

1.4 Outline

In Chapter 2, we discuss some issues arising from the modeling process which appears in simulation and fuzzy set literature. Here, we address the importance of

the expert's role for validating simulation models and for constructing fuzzy models. Then, we review the fuzzy set theory that is relevant to this research and its relation to computer simulation. First, we explain the basic definitions of fuzzy set theory and various approaches for constructing fuzzy membership functions. Since one of the most successful application areas of fuzzy systems is the fuzzy controller, we present its overview with its key components. Finally, we briefly discuss the fuzzy simulation approaches which appear in both fuzzy set and computer simulation literature.

In Chapter 3, we present a new fuzzy simulation approach. We first assume three forms of experts' rules as the inputs to the fuzzy simulation. Then, we propose three different fuzzy simulation algorithms for directly encoding the fuzziness in the expert rules into computer simulation.

In Chapter 4, based on the fuzzy simulation method discussed in Chapter 3, we describe an interactive environment that we've developed for isolating inconsistent knowledge. First, we introduce two measurements of inconsistency that we've employed for quantitative comparison. Then, we provide two consistency checking procedures to handle various forms of experts' estimates on linguistic rules. Finally, we analyze the time complexities of these procedures.

In Chapter 5 and Chapter 6, we consider a steam-powered propulsion ship (FULTON) and a predator-prey population, respectively, to illustrate the applications of the methodology. Finally, future work and conclusion are presented in Chapter 7 and Chapter 8, respectively.

CHAPTER 2 BACKGROUND AND RELATED WORK

2.1 Modeling Process in Simulation and Fuzzy Set Literature

2.1.1 Simulation Modeling Process

Figure 2.1 [44, 45, 7] shows the general simulation modeling process and its relation to validation and verification. The *problem entity* is the real or proposed system to be modeled. The *conceptual model* represents the mathematical, logical or verbal representation of the problem entity, and this model is developed through an analysis or modeling phase. The *computerized model* represents the conceptual model implemented on a computer, which is developed through a computer programming and implementation phase. Inference about the problem entity is obtained by conducting computer experiments on the computerized model in the experimentation phase.

The *conceptual model validity* determines the validity of the underlying assumptions and theories by using mathematical or statistical methods. Additionally, the process is concerned with whether this specific model's representation of the problem entity being modeled and its structure, logic and mathematical and causal relationships are *reasonable* for the intended use of the model [44]. For such a process, techniques such as *face validation* and *traces* are used. *Face validation* involves having domain experts evaluate the conceptual model to determine if they believe it is correct and reasonable for its purpose. This usually means examining the flowchart, graphical model or the set of model equations. In the *traces* the behavior of different types of specific entities in the model are traced through the model to determine if the model's logic is correct and if the necessary accuracy is obtained.

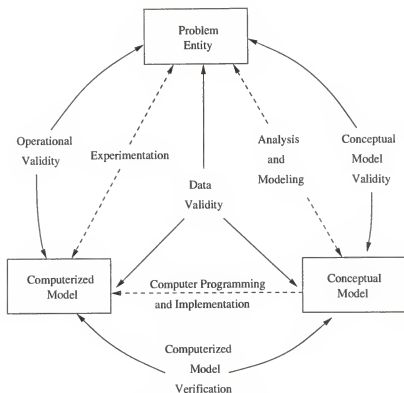


Figure 2.1. Modeling process and its relation to validation/verification

Computerized model verification determines that the implementation of the conceptual model is correct. The verification techniques used here can be found in the software engineering field, such as formal correctness proof, structured walk-through, top-down and bottom-up testing [47].

Operational validity is defined as determining that the model output has sufficient accuracy for its intended purpose. Most of the validation efforts take place in this stage. Any discrepancy found may be due to an inadequate conceptual model or an improperly implemented conceptual model. In this case, one set of system data is used for calibration of the simulation model, and another independent set is used for validation. If the simulation output data agree with the system output data, the model can be considered valid. For this reason, *data validity* is needed for comparing

the problem entity's behavior with the model's behavior, as well as for building the conceptual model, for developing theories and for testing the underlying assumption.

However, in determining *operational validity*, it is difficult to use the classical statistical test (t, two-sample chi-square, etc.) between the model output and the corresponding system output data, due to the nature of the data [29]. Specifically,

- an observation may be *nonstationary*: the distribution of the successive observations change over time.
- an observation may be *autocorrelated*: the observations are correlated with each other.

Therefore, techniques commonly used for operational validity are [45]

- *face validation*, where experts are asked to make subjective judgements on whether the model has sufficient accuracy,
- statistical tests for *confidence intervals* and *hypotheses* and
- *turing test*, where individuals knowledgeable about the system are asked if they can discriminate between system and the model outputs.

However, the statistical methods mentioned above also have difficulties in some applications, such as military or manufacturing systems, due to the paucity of real-world data. Because of the difficulty in obtaining *data validity*, human knowledge about the system takes a relatively important role during the entire validation process, where approaches such as comparison to other models on graphical displays, intuition, opinions or past experience are usefully adopted [6]. The importance of human information and knowledge representation in problem solving tasks are discussed in the field of Systems Engineering [43]. Possibilities for expert aids in model validation are presented in [44].

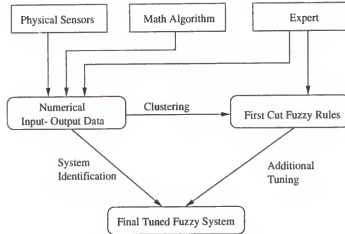


Figure 2.2. Fuzzy engineering

2.1.2 Fuzzy Modeling Process

Fuzzy engineering [28] is an emerging terminology which refers to the overall trials for finding the good qualities of the fuzzy system as a function approximator for a given physical systems. The basic unit of the fuzzy function approximation is the *IF-THEN* rule, and the quality of the approximation depends on how smart the rules are. For example, a few rules give a quick approximation, but are less accurate and which make the tuning process harder thereafter. On the other hand, constructing rules covering all input-output state spaces defined by the Cartesian product of the input pattern and the output pattern easily causes the rule explosion. For these reason, fuzzy engineering mainly concerns itself with developing efficient tuning algorithms or finding optimal set of rules.

Figure 2.2 shows a general process for modeling fuzzy systems. The modeling process can start with numerical input-output data. Fuzzy system identification, often called *black box modeling*, is the most common approach for designing mathematical models of dynamic systems from numerical input-output data. The black box modeling contrasts with the *white box modeling*, where mathematical models are driven from already known physical laws [32]. The derivation of black box fuzzy models

from numerical data employs various clustering methods, neural networks and genetic algorithms to identify fuzzy rules with their membership functions. The recent achievements and comprehensive discussion of the black box modeling can be found in the literature [23, 36]. Their main tasks are structure and parameter estimations. However, the fundamental limitation in using numerical data comes from the *curse of dimensionality* problem [32]. That is, the number of possible rules, n , increases exponentially with the number of possible variables, r by the formula

$$n = \prod_{k=1}^r n_k, \quad (2.1)$$

where n_k is the number of fuzzy values in the k th variable. Since all variables are not important to describe the behavior of the underlying system, searching for influential variables among all possible variables causes the problem of combinatorial optimization. Therefore, the modeling process using these techniques suffers from computational complexity or local optimality [28, 32].

In contrast, the modeling process can also start with a *first cut* fuzzy system as shown in Figure 2.2. In practice, the first cut fuzzy system can be directly obtained from the expert in the form of a verbal expression. The domain expert forms rules by performing the following three steps [28]: (1) selecting input and output fuzzy variables, (2) selecting fuzzy values of these variables and (3) constructing fuzzy rules by relating the input values to the output values. Rough fuzzy rules also can be obtained from numerical input-output data through unsupervised learning by clustering the data; this gives quick approximation, but is less accurate.

The utilization of prior knowledge provided by experts alleviates many problems arising from using traditional system identification, because important variables of the system and their approximate forms of values are provided by the experts. Another advantage of utilizing the expert's prior knowledge is that some system phenomena such as those cannot be revealed by means of collecting the physically observed

input-output data can be captured. However, the first potential drawback of using the expert's knowledge is that it is difficult for the human expert to capture all causal system relations, especially in modeling complex systems. Even the expert is often unaware of all of the contextually dependent factors qualifying generalizations [13]. Secondly, obtaining such a good quality of first cut fuzzy rules is not easy in practice. Generally speaking, the gap between the implementation level (i.e., fuzzy rules) and the expert's knowledge is too wide. Specifically, both *linguistic imprecision* and *uncertainty* may exist in a fuzzy rule [22]. For example, given a fuzzy rule,

IF wind is *high* THEN the sailing should be good (0.8),

the linguistic imprecision comes from the fuzzy value "*high*" and the uncertainty comes from the numeric value (often called *confidence factor*) "0.8." The linguistic imprecision results from linguistic inexactness, specially, the linguistic vagueness in which the boundary is not clearly defined [22, 62]. Uncertainty can arise because reliable information cannot always be gathered to assign a probabilistic uncertainty; moreover, even an expert may be unsure of a particular piece of causal information. Much of the uncertainty in such cases is possibilistic rather than probabilistic in nature [60].

2.2 Fuzzy Set Theory and Its Application

This section presents a review of the relevant aspects of fuzzy set theory which form the basis of our fuzzy simulation. The theory of *fuzzy sets* can be found in References [57, 58, 59, 27, 14, 61]. Generally speaking, fuzzy sets may be viewed as an attempt to deal with a type of imprecision arising when the boundaries of classes are not sharply defined. A fuzzy set A of a universe of discourse X is characterized by a membership function $\mu_A : X \rightarrow [0, 1]$ which associates with each element x of

X a number $\mu_A(x)$ in the interval $[0, 1]$ which represents the grade of membership of x in A .

2.2.1 Notation, Terminology, and Basic Operations

- *Definition 2.1:* A fuzzy set A of the universe of discourse X is *convex* if and only if for all x_1, x_2 in X

$$\mu_A(\lambda x_1 + (1 - \lambda)x_2) \geq \min(\mu_A(x_1), \mu_A(x_2)),$$

where $\lambda \in [0, 1]$.

- *Definition 2.2:* A fuzzy set A of the universe of discourse X is called a *normal* fuzzy set if $\exists x_i \in X, \mu_A(x_i) = 1$.
- *Definition 2.3:* A *fuzzy number* is a fuzzy set in the universe of discourse X that is both convex and normal.

To simplify the representation of fuzzy sets, a finite fuzzy set, A , of X is expressed as

$$A = \mu_A(x_1)/x_1 + \mu_A(x_2)/x_2 + \dots + \mu_A(x_n)/x_n, \text{ or } A = \sum_{i=1}^n \mu_A(x_i)/x_i,$$

where $+$ sign denotes the union rather than the arithmetic sum.

If the fuzzy set, A , is not finite, A may be represented in the form $A = \int_X \mu_A(x)/x$ in which the integral sign stands for the union of the fuzzy singletons $\mu_A(x)/x$.

- *Definition 2.4:* The height of fuzzy set A , $height_A$, in the universe of discourse X is the supremum of $\mu_A(x)$ over A . Formally,

$$height_A = \sup_{x \in X} \mu_A(x). \quad (2.2)$$

- *Definition 2.5:* The *complement* of A is denoted by \bar{A} and is defined by

$$\bar{A} = \int_X (1 - \mu_A(x))/x. \quad (2.3)$$

The operation of complementation corresponds to negation.

- *Definition 2.6:* The union of fuzzy sets A and B is denoted by $A \cup B$ and is defined by

$$A \cup B = \int_X (\mu_A(x) \vee \mu_B(x)) / x, \quad (2.4)$$

where \vee is a maximum operator.

- *Definition 2.7:* The intersection of fuzzy set A and B is denoted by $A \cap B$ and is defined by

$$A \cap B = \int_X (\mu_A(x) \wedge \mu_B(x)) / x, \quad (2.5)$$

where \wedge is a minimum operator.

- *Definition 2.8:* If f is an n -ary crisp function which is a mapping from a Cartesian product $X_1 \times \cdots \times X_n$ to a space Y , and if A is a fuzzy set in $X_1 \times \cdots \times X_n$ which is characterized by a membership function $\mu_A(x_1, \dots, x_n)$, with $x_i, i = 1, \dots, n$, denoting a generic point in X_i , then *extension principle* [59] says

$$\begin{aligned} f(A) &= f\left(\int_{X_1 \times \cdots \times X_n} \mu_A(x_1, \dots, x_n) / (x_1, \dots, x_n)\right) \\ &= \int_Y \mu_A(x_1, \dots, x_n) / f(x_1, \dots, x_n). \end{aligned} \quad (2.6)$$

It is assumed that the membership function of A is expressed by

$$\mu_A(x_1, \dots, x_n) = \mu_{A_1}(x_1) \wedge \mu_{A_2}(x_2) \wedge \cdots \wedge \mu_{A_n}(x_n), \quad (2.7)$$

where $\mu_{A_i}, i = 1, \dots, n$, is the membership function of A_i .

- *Definition 2.9:* Let A and B represent two fuzzy numbers and let \star denote any of the four basic arithmetic operations. Then, using the extension principle

(2.6) under the assumption (2.7), we define fuzzy set, $A \star B$ on \mathcal{R} , where \mathcal{R} is a set of all real numbers, as

$$\mu_{A \star B}(z) = \max_{z=x \star y} (\mu_A(x) \wedge \mu_B(y)), \quad (2.8)$$

for all $z \in \mathcal{R}$. Thus, for example, if $A, B \subseteq \mathcal{R}$ are two fuzzy numbers with respective membership functions $\mu_A(x)$ and $\mu_B(y)$, then the four basic arithmetic operations, i.e., addition, subtraction, multiplication and division give, for each $x, y, z \in \mathcal{R}$, the following results:

$$\mu_{A+B}(z) = \max_{z=x+y} (\mu_A(x) \wedge \mu_B(y)), \quad (2.9)$$

$$\mu_{A-B}(z) = \max_{z=x-y} (\mu_A(x) \wedge \mu_B(y)), \quad (2.10)$$

$$\mu_{A \times B}(z) = \max_{z=x \times y} (\mu_A(x) \wedge \mu_B(y)), \quad (2.11)$$

$$\mu_{A \div B}(z) = \max_{z=x \div y} (\mu_A(x) \wedge \mu_B(y)). \quad (2.12)$$

- *Definition 2.10:* Let P be a compound statement of the type

$$(\mathcal{X} \text{ is } A) \star (\mathcal{Y} \text{ is } B),$$

where

\mathcal{X}, \mathcal{Y} = fuzzy variables that take real numbers from some universal set X, Y , respectively,

A, B = fuzzy values on X, Y , respectively, and

\star = a conjunction (and) or a disjunction (or).

When \star is a conjunction, the *rule of conjunctive composition* [61] states that P can be expressed by a possibility distribution $\pi(x, y)$ which is defined by

$$\{\mu_{A \times B}(x, y) / (x, y) \mid x \in X, y \in Y\}, \quad (2.13)$$

where

$\mu_{A \times B}(x, y) = \min(\mu_A(x), \mu_B(y))$, and \times = Cartesian product.

When $*$ is a disjunction, the *rule of disjunctive composition* [61] states that P can be expressed by a possibility distribution $\pi(x, y)$ which is defined by Equation (2.13), where $\mu_{A \times B}(x, y) = \max(\mu_A(x), \mu_B(y))$.

- *Definition 2.11:* A *fuzzy relation* [58] R from a set X to a set Y is a fuzzy subset of the Cartesian product $X \times Y$. R is characterized by a bivariate membership function $\mu_R(x, y)$ and is expressed

$$R = \int_{X \times Y} \mu_R(x, y)/(x, y). \quad (2.14)$$

- *Definition 2.12:* A *fuzzy conditional statement*, IF $\mathcal{X} = A$ THEN $\mathcal{Y} = B$, or for short, $A \Rightarrow B$, in which A and B are fuzzy sets, can be defined as a fuzzy relation $A \times B$.
- *Definition 2.13:* If R is a relation from X to Y and S is a relation from Y to Z , then the *max-min composition* [58] of R and S is a fuzzy relation denoted by $R \circ S$ and defined by

$$R \circ S = \int_{X \times Z} \bigvee_y (\mu_R(x, y) \wedge \mu_S(y, z))/(x, z), \quad (2.15)$$

where \bigvee_y is the supremum over the domain of y .

- *Definition 2.14:* In traditional logic, one of the main tool for reasoning is *modus ponens*, that is, $(A \wedge (A \Rightarrow B)) \Rightarrow B$. Then an approximate extension of the *modus ponens*, called *generalized modus ponens* can be used for *approximate reasoning* in the following type of inference

Premise: \mathcal{X} is A'

Implication: IF \mathcal{X} is A THEN \mathcal{Y} is B

Conclusion: \mathcal{Y} is B' ,

where the conclusion B' can be derived by *min-max composition rule of inference*

$$B' = A' \circ [A \times B]. \quad (2.16)$$

2.2.2 Membership Function Construction

Numerous methods for constructing membership functions have been described in the literature [27, 42, 41, 9, 52, 2, 3, 33, 19]. All these methods can be classified into two approaches: *direct* and *indirect* methods. Both methods are further classified depending on whether one expert or multiple experts are involved.

In the *direct* method [27], given a fuzzy set A , an expert is expected to assign a membership degree, $\mu_A(x)$, to each element x , according to his or her opinion. This can be done by defining a complete membership function in terms of mathematical formula, or by exemplifying it by answering a question such as “what is the degree of membership of x in A ?”

In the *indirect* method, an expert is required to answer simpler questions which are easier to answer and less sensitive to the various biases of subjective judgement. The most common approach is based on pairwise comparisons [42, 41] of relevant elements, which replaces the direct estimates of membership degree. An example of typical question in this method is “which color, A or B , has the property of darkness more strongly, and how much more?” For all pairs of elements, comparisons are repeated by giving numerical scale to express the relative strength of the property. The relative weights are represented by a nonsymmetric full matrix. Then the membership degrees are the components of the eigenvector corresponding to the maximum eigenvalue.

In this section, we briefly review basic methods for constructing membership functions from sample data gathered from the *direct* or *indirect* method. These include the *curve-fitting method* and *learning from neural network*. In these methods,

we assume that n sample data

$$\langle x_1, a_1 \rangle, \langle x_2, a_2 \rangle, \dots, \langle x_n, a_n \rangle \quad (2.17)$$

are given, where $x_i, i = 1, 2, \dots, n$ is a real number and a_i is the membership degree of x_i in a fuzzy set A .

Lagrange Interpolation

Lagrange interpolation is a curve-fitting method in which the constructed membership function is represented by a polynomial form defined by

$$f(x) = a_1 L_1(x) + a_2 L_2(x) + \dots + a_n L_n(x), \quad (2.18)$$

where

$$L_i(x) = \frac{(x - a_1) \dots (x - a_{i-1})(x - a_{i+1}) \dots (x - a_n)}{(x_i - a_1) \dots (x_i - a_{i-1})(x_i - a_{i+1}) \dots (x_i - a_n)}. \quad (2.19)$$

Since values $f(x)$ need not be in $[0,1]$, the following formula is applied to make the fuzzy set A normal:

$$\mu_A(x) = \max[0, \min[1, f(x)]]. \quad (2.20)$$

Even though the membership function matches the sample data exactly, complexity increases with the number of sample data. Besides, for the values of x outside the given sample range, this method does not work well. This requires that sample data be well distributed over the fuzzy set A .

Least-square Curve Fitting

Given sample data (2.17) and a suitable function $f(x; \alpha, \beta, \dots)$, where α, β, \dots are parameters whose values distinguish function in its class from one another, this method selects a function $f(x; \alpha_0, \beta_0, \dots)$ from the class for which

$$E = \sum_{i=1}^n [f(x_i; \alpha, \beta, \dots) - a_i]^2 \quad (2.21)$$

reaches its minimum. Then, we apply Equation (2.20) to make a normal fuzzy set. Such a suitable function is chosen from standard distributions based on expert's experience or experimental comparison with other classes. The bell-shaped function defined below has been frequently used for such a purpose.

$$f(x; \alpha, \beta, \gamma) = \gamma e^{-(x-\alpha)^2/\beta}, \quad (2.22)$$

where α is a location for center, $\sqrt{\beta/2}$ defines the inflection points, and γ is height parameter.

Learning from Neural Networks

The literature dealing with the use of neural networks for constructing membership functions is rapidly growing [9, 52, 2, 3]. Construction of such functions are done by learning patterns from sample data defined in (2.17). Let each input of x be x^p , its expected output be t^p and its actual output be y^p . Given suitable hidden layers and activation function, by initializing weights of the network and applying pairs $\langle x^p, t^p \rangle$ of the training set to the neural network, we can calculate the *square error*:

$$E_p = \frac{1}{2}(y^p - t^p)^2. \quad (2.23)$$

Then, to minimize the E_p , we update weights according to backpropagation algorithm. At the end of each cycle, a *cumulative error* defined as

$$E = \frac{1}{2} \sum_{p=1}^n (y^p - t^p)^2 \quad (2.24)$$

is compared against E_{max} specified by user. A new cycle is initiated until $E \leq E_{max}$. When $E \leq E_{max}$, the desired membership function is obtained.

2.2.3 Fuzzy Controller

A fuzzy system is any system whose variables range over states that are fuzzy sets. The most successful application area of fuzzy system has been the area of *fuzzy*

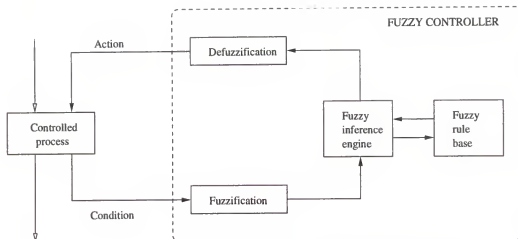


Figure 2.3. A general architecture of a fuzzy controller

control. Fuzzy controllers are special expert systems in the sense that each has a knowledge base represented by fuzzy inference rules and an inference engine. Fuzzy controllers are capable of utilizing knowledge elicited from human operators in control problems when [10]:

- one or more of the control variables are continuous,
- a mathematical model of the process does not exist, or exists but is too difficult to encode, and
- a mathematical model of the process is too complex to be evaluated quickly enough for real-time operation.

In those cases, an imprecise linguistic description consisting of a set of control rules can usually be articulated by the human operators with relative ease. Figure 2.3 shows the architecture of a fuzzy controller.

In this section, we discuss the design process of a fuzzy controller. Let us consider a very simple fuzzy controller as shown in Figure 2.4. In this figure, the controller monitors two control variables, e and \dot{e} , where e is defined as an error between the actual value of the controlling variable v and its desired value, and \dot{e} denotes the rate

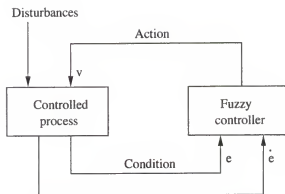


Figure 2.4. A simple fuzzy controller

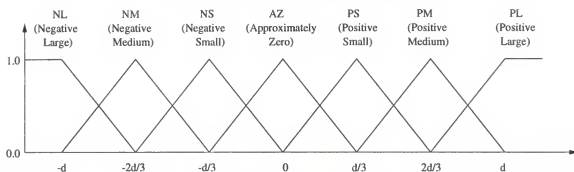


Figure 2.5. A fuzzy quantization

of change in the error. Depending on e and \dot{e} , relevant control actions represented by v are produced.

Step 1: Fuzzy Quantization

The first step is to obtain fuzzy quantizations by identifying relevant input and output variables and their ranges and selecting appropriate labels (i.e., fuzzy sets) for each variable. The number of labels associated with a variable is generally an odd number between 5 and 9 [10]. For the reason of symmetry, an odd number is preferred. The number of labels determines the *expressiveness* and the *predictiveness* of the fuzzy system [31]. The *expressiveness* is a measure for the information content that the model provides, while the *predictiveness* is a measure for its forecasting power. Since these two measures are contradictory, we should compromise. Some observations have been reported in which either three or five labels were about optimal in most

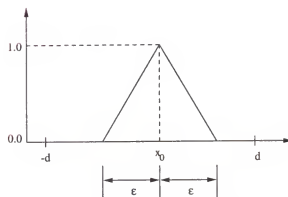


Figure 2.6. A fuzzified measurement

practical applications [8, 53]. Figure 2.5 shows an example of the quantization where triangular fuzzy numbers that are equally spread over the range $[-d, d]$ are used.

Step 2: Fuzzification

In this step, a fuzzification function f for each input variable is chosen to express measurement uncertainty. For example, by applying an appropriate fuzzification function, a measurement $e = x_0$ can be defined by a fuzzy set as shown in Figure 2.6, where ϵ is determined in the context of each particular application. This fuzzy set acts as a fact in the inference process (Step 4).

Step 3: Obtain Conceptual Model

In this step, a conceptual model is obtained in terms of a set of fuzzy inference rules that describe the action taken on each combination of control variables. Two common ways for obtaining such information are: 1) from human operators or 2) from empirical data by suitable learning methods [31, 51, 38, 50, 49]. The canonical form of the inference rule is

IF e is A and \dot{e} is B THEN v is C .

Generally, the number of rules required depends on the number of control variables [10]. For example, if a fuzzy controller requires n control variables and m fuzzy regions for each variable, the system generally requires m^n rules for a total of m^n

		e								
v		NL	NM	NS	AZ	PS	PM	PL		
e	NL	PL				PM	AZ			
	NM									
	NS	PM		PM	PS	AZ	NM			
	AZ			PS	AZ	NS				
	PS			AZ	NS	NM				
	PM	AZ		NM	NL					
	PL									

Figure 2.7. An example of fuzzy inference rules

possible input combinations. Since the number of fuzzy rules grows exponentially with the number of system variables, the search for optimal rules forms one of the main research areas of fuzzy engineering. However, most applications so far have had few inputs and outputs, and this has helped keep the rule explosion manageable [28]. Figure 2.7 shows an example fuzzy rule base in fuzzy sets defined in Figure 2.5.

Step 4: Design an Inference Engine

Here, to determine the resulting fuzzy set in multiconditional approximate reasoning represented by the form

Rule 1: IF e is A_1 and \dot{e} is B_1 THEN v is C_1

Rule 2: IF e is A_2 and \dot{e} is B_2 THEN v is C_2

.....

Rule n : IF e is A_n and \dot{e} is B_n THEN v is C_n

Fact: e is A' and \dot{e} is B'

Conclusion: v is C' ,

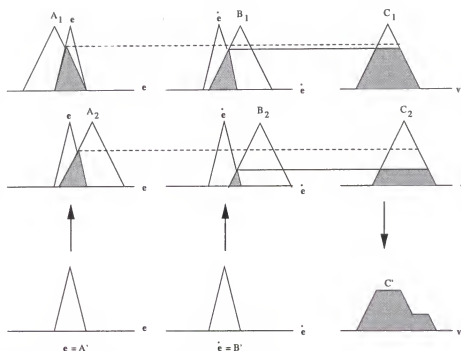


Figure 2.8. An illustration of the *min-max composition rule of inference*

a method called *min-max composition rule of inference* defined in *Definition 2.14* is commonly applied. That is, through *min inference*, each output membership function is cut off at a height corresponding to the minimum degree of the truth of the rule premise. Then, through *max composition*, a combined output membership function is constructed by taking pointwise maximums over all of the fuzzy set assigned to the output variable. An illustration of the method for two fuzzy rules is given in Figure 2.8.

Step 5: Defuzzification

This step performs defuzzification by converting output fuzzy set into a single real number. Two most common methods for defuzzification are *centroid* and *composite maximum*. The *centroid* method takes the center of gravity of the output fuzzy set C' in Figure 2.8. The defuzzified value d is calculated by the formula

$$d = \frac{\sum_{i=1}^n \mu_{C'}(v_i) \times v_i}{\sum_{i=1}^n \mu_{C'}(v_i)}, \quad (2.25)$$

where $v_i, i = 1...n$ is an element in fuzzy set C' .

In the *composite maximum* method, a defuzzified value d is defined as the average of the smallest value and the largest value of v for which $\mu_{C'}(v)$ is the height of C' , $height_{C'}$. Formally,

$$d = \frac{\min\{v_i \mid v_i \in M\} + \max\{v_i \mid v_i \in M\}}{2}, \quad (2.26)$$

where $M = \{v_i \mid \mu_{C'}(v_i) = height_{C'}\}$.

2.3 Fuzzy Set Theory in Computer Simulation

Probability-based methods are useful when most of the uncertainty can be effectively described through the use of large data sets and their associated moments. However, experts often do not think in probability values, but in terms such as *much, usually, always, sometimes*, etc. In domains where estimation or measurement of probabilities is not amenable, fuzzy set theory offers an alternative [22]. Here, we can use any type of fuzzy number, such as an interval-valued fuzzy number, a triangular fuzzy number, a trapezoidal fuzzy number or a general discrete (or continuous) fuzzy number, depending on the degree of uncertainty.

Owing to the *extension principle* [61] in the fuzzy set theory, nonfuzzy mathematical structures can be made fuzzy. Here is a sample of how this relates to simulation. The following can be made fuzzy [58, 18]: 1) a state variable value including initial conditions, 2) parameter values, 3) inputs and outputs, 4) model structures and 5) algorithmic structures. For example, we can use fuzzy simulation to execute a fuzzy automaton, where its states are characterized by fuzzy sets, and the production of responses and the next states are facilitated by appropriate fuzzy relations [27]. For another example, a fuzzy algorithm, defined as a ordered set of fuzzy instructions, can be used to provide an approximate analysis of systems and decision processes

that are too complex for the application of conventional mathematical techniques [58].

Three kinds of fuzzy simulation approaches have been reported in the simulation literature: *Qualitative Simulators* (i.e., *Qua.Si* [4]), *Fuzzy Qualitative Simulation* (i.e., *Fusim* [46]) and *Correlated Uncertainty method* [15, 16, 17]. The *Qua.Si* and the *Fusim* are useful for qualitative simulations where simulations are performed using fuzzy sets themselves based on fuzzy calculus or fuzzy arithmetic. The third approach takes fuzzy sets from experts and, through deterministic sampling from fuzzy sets, it performs computer simulation *quantitatively* on discrete event or continuous models using real arithmetic. For such a process, every vertex in the fuzzy number is issued independently to the simulation function, and the outputs of the simulation are mapped into the most closely matched fuzzy linguistic value by a linguistic approximation. Thus, rules can be extracted, and these results can be validated against the expert's domain knowledge. The fuzzy simulation method that we've employed for isolating inconsistency is an extended version of the *correlated uncertainty method*.

The algorithm for fuzzy simulation using the correlated uncertainty method is [17]:

1. Let a fuzzy simulation component such as a parameter \mathbf{p} be defined as a triangular fuzzy number F , where $F = (a, b, c)$. Assume the fuzzy number is identified by brackets (i.e., $F[2] = b$).
2. For $j \in 1, 2, 3$:
 - (a) Let $\mathbf{p}[j] = F[j]$.
 - (b) SIMULATE REAL
 - (c) $\forall i$, obtain $x_i(t_e)[j]$,

(a)	day	interarrival rate	service rate
	Saturday Morning	short	fast
	Saturday Day	short	medium
	Saturday Evening	medium	medium

	Tuesday Morning	long	slow

(b)	line size	customer satisfaction
	short	happy
	medium	complacent
	long	irritated

(c)	customer satisfaction	grocery sales
	happy	good
	complacent	good
	irritated	bad

(d)	day	line size	customer satisfaction
	Saturday morning	short	happy

(e)	day	service rate	customer satisfaction
	Tuesday morning	slow	irritated

Figure 2.9. Information presented by an expert

where

SIMULATE REAL denotes simulation using real arithmetic instead of fuzzy arithmetic,

t_e = the end time for the simulation, and

x_i = the state variables of interest.

Because of the SIMULATE REAL, the simulation is accomplished by performing multiple simulations; the number of simulations depends on the order of the fuzzy numbers. The outputs of a simulation can be mapped into the most closely matched fuzzy linguistic values by using a *distance metric*. Let the simulation outputs of m ordered fuzzy number be defined as $F(1), F(2), \dots, F(m)$, and let the possible n output linguistic values be q_1, q_2, \dots, q_n . Then the distance metric is defined by

$$\text{Min} \sum_{i=1}^m |F(i) - q_j(i)|, \quad (2.27)$$

where $j = 1, 2, \dots, n$.

Let's take a simple grocery store example [16] to illustrate the concept of the fuzzy simulation. We assume that an employee (cashier) is an expert, and he or she provides the information represented as in Figure 2.9. Note that the *line size*

affecting the *customer satisfaction* is again influenced by both the *interarrival rate* and the *service rate*. However, as in this example, the expert may not be able to represent such complex relations precisely. Here, fuzzy simulation comes into play. By replacing those grocery store statistics with the compiled knowledge of queuing models, we can identify the deep knowledge for such complex relations as a hypothesis of the expert's knowledge.

A single server queue as a deep model representing a grocery check out line can be associated with the following pseudo-code:

```

schedule an arrival now;
while (not end of simulation) do
    get_next_event;
    switch on <event>:
        ARRIVAL: schedule REQUEST_SERVER now;
                schedule ARRIVAL using fuzzy arrival time;
        REQUEST_SERVER: if server is free then
                        schedule RELEASE_SERVER using fuzzy service time;
                        else
                            queue customer;
        RELEASE_SERVER: release server to next customer;
    endwhile

```

By executing the fuzzy simulation, we obtained a total of twelve relations between *interarrival rate*, *service rate* and *customer satisfaction*. The entire processes are shown in Figure 2.10. Note that, as with any good analysis, the analyst must ensure that the fuzzy number definitions agree as closely as possible with the expert that issued those definitions through the usual knowledge acquisition procedure where the fuzzy knowledge is first elicited.

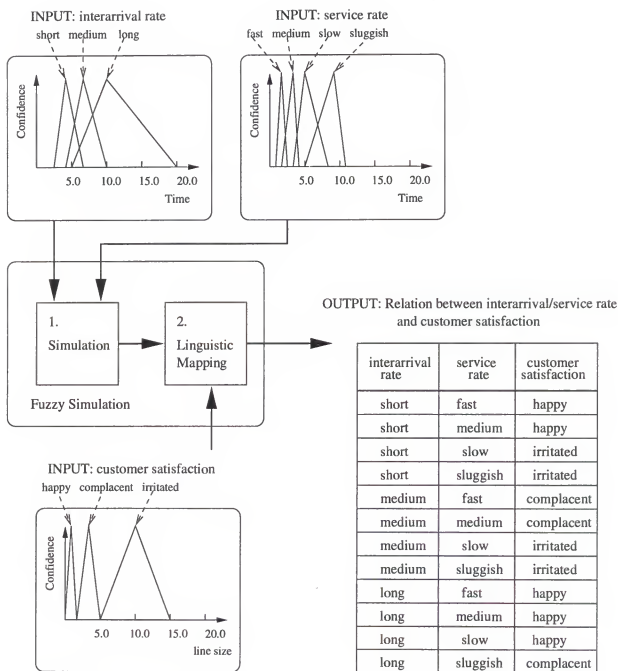


Figure 2.10. Applying fuzzy simulation to grocery store example

By executing the fuzzy simulation in Figure 2.10, we obtain the quantitative plot of *line size* over time based on each combination of three linguistic values for the *interarrival rate* and four linguistic values for the *service rate*. Sampling is based on the three vertices of the fuzzy number being used to define both the *interarrival rate* and the *service rate*. From this, we obtain a time-variant description of customer satisfaction over time. This is done by mapping from data for the line size into the fuzzy linguistic values for *customer satisfaction* using the distance metric.

Output of the fuzzy simulation shown in Figure 2.10 forms a more complete rule base as a hypothesis of the expert knowledge. When we compare them to Figure 2.9 (a) and (e), we can note that there is a conflict between the rules concerning Tuesday morning. The queuing model predicted a *happy* customer, whereas the expert specified an *irritated* customer. When such conflicts arise, the expert can either reevaluate his original rule as *slightly off* or a set of parameters can be changed in the queuing model. In this way, the expert must evaluate the new rules created by the fuzzy simulation to see if there is agreement with his expertise.

In this chapter, we reviewed the background that is relevant to this research. We discussed the simulation and the fuzzy modeling processes in general and addressed the importance of the expert's role during model validation processes. In the next chapter, we present a new fuzzy simulation approach to bridge the gap between the expert rules and an assumed quantitative model. The fuzzy simulation approach handles, particularly, the *possiblistic uncertainty* in the expert's rules by directly encoding the uncertainty into the simulation components.

CHAPTER 3

A NEW FUZZY SIMULATION APPROACH

The fuzzy simulation approach introduced here has extended the original version of *correlated uncertainty method* discussed in Chapter 2. In this extended approach, by carrying membership degrees of fuzzy sets in the expert's rule premise and issuing them to simulation components, we are able to calculate a confidence factor for each rule, which is then compared against the confidence factor of the expert rule. This quantitative measurement provides us with useful information such as "which is the most inconsistent rule?" and "how consistent are two given rule sets?" This facility plays a basic role for isolating inconsistent knowledge through interactive user control. In the following sections, we assume three types of expert rules as the inputs to the fuzzy simulation. Then, we present how to handle these types of rules differently using the fuzzy simulation approach.

3.1 Expert Rule Format

The input of fuzzy simulation is a collection of expert rules. In what follows, we assume that the three following canonical forms of rules are presented by experts.

- IF \mathcal{X} is A THEN \mathcal{Y} is B (CF),
- IF \mathcal{X} is $(A_1 \star A_2)$ THEN \mathcal{Y} is B (CF) and
- IF $(\mathcal{X} \text{ is } A) * (\mathcal{Y} \text{ is } B)$ THEN \mathcal{Z} is C (CF),

where

\mathcal{X} , \mathcal{Y} and \mathcal{Z} = *fuzzy variables* that take real numbers from some universal sets, X , Y and Z , respectively,

Table 3.1. Notation

Notation	Usage
$RULE_{expert}$	<i>Rules presented by expert</i>
$RULE_{fuzzy}$	<i>Rules generated by fuzzy simulation</i>
MF_{expert}	<i>Fuzzy Membership Functions presented by an expert</i>
MF_{fuzzy}	<i>Fuzzy Membership Functions generated by fuzzy simulation</i>
$MF_{premise}$	<i>Membership Functions of fuzzy value in rule premise</i>
MF_{conseq}	<i>Membership Functions of fuzzy value in rule consequence</i>
CF_{expert}	<i>Confidence Factor presented by an expert</i>
CF_{fuzzy}	<i>Confidence Factor calculated by fuzzy simulation</i>

A , A_1 and A_2 = fuzzy values on X ,

B and C = fuzzy values on Y and Z , respectively,

CF = a confidence factor in the rule consequence given that the premise conditions are satisfied, and

\star and \ast = arithmetic ($+$, $-$, \times or \div) and logic (*or* or *and*) operator, respectively.

We call the first type of rule a *simplex rule*, and the other type a *compound rule*.

The following is an example of the compound rule with the logic operator *and*:

IF (Temperature is High) and (Pressure is Slightly_Low)

THEN Heat_Change should be Slightly_Negative ($CF = 0.8$).

The premise parts of the last two canonical types of rules can be combined to make a more complex rule such as

IF (\mathcal{X} is ($A_1 + A_2$)) or (\mathcal{Y} is ($B_1 + B_2$)) THEN \mathcal{Z} is C .

For simplicity, the notation in Table 3.1 will be used in the entire chapter.

3.2 Fuzzy Simulation

The fuzzy simulation method introduced here is capable of simulating the expert rules using quantitative models. For each expert rule, this method takes the premise

part and its $MF_{premise}$, and through simulation, it generates a conclusion with a CF_{fuzzy} . With the intention of comparing this result directly against the expert rule, the fuzzy simulation method is forced to derive a conclusion with the same linguistic value that the expert presented, but with possibly a different CF_{fuzzy} from the CF_{expert} .

When the expert rule is *simplex*, fuzzy simulation involves one simulation for each element within a $MF_{premise}$. In contrast, when the rule is *compound*, we first obtain an intermediate fuzzy set by applying the *extension principle* defined in *Definition 2.8* or the *rule of conjunctive or disjunctive composition* defined in *Definition 2.10*, depending on whether the operator type in the premise is arithmetic or logic. Fuzzy simulation using the compound rules involves one simulation for each element within the intermediate fuzzy set. Finally, we calculate a CF_{fuzzy} for that rule using a *weighted average method*.

3.2.1 Fuzzy Simulation for Simplex Rules

Consider a simplex rule of the type

IF \mathcal{X} is A THEN \mathcal{Y} is B ,

where

\mathcal{X}, \mathcal{Y} = fuzzy variables that take real numbers from universal set X, Y , respectively, and A, B = fuzzy values on X, Y , respectively.

Algorithm of Fuzzy Simulation

1. Let a fuzzy simulation component such as a parameter \mathbf{p} be defined as a fuzzy set A , where

$$A = \mu_A(x_1)/x_1 + \mu_A(x_2)/x_2 + \dots + \mu_A(x_n)/x_n.$$

Assume the element of A is identified by brackets (i.e., $A[2] = x_2$).

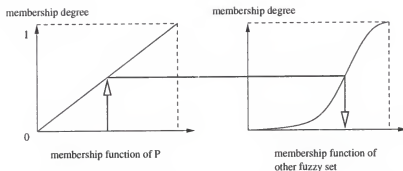


Figure 3.1. A two-step process of searching membership degree

2. For $j \in 1, 2, \dots, n$:

- (a) Let $\mathbf{p}[j] = A[j]$.
- (b) SIMULATE REAL.
- (c) obtain $(\mu_B(y_j)/y_j)(t_e)$.

3. calculate CF_{fuzzy} ,

where

SIMULATE REAL denotes simulation using real arithmetic,

$y_j, j = 1, \dots, n$ denotes real values on Y , and

t_e = the end time for the simulation.

During SIMULATE REAL, the correlated uncertainty method requires that when we replace \mathbf{p} with a real number whose membership degree is d , we should replace other fuzzy simulation components with real numbers whose membership degrees are also d . This procedure involves a two-step process of searching for the membership degree of \mathbf{p} , and then using this degree to drive the elements of other fuzzy sets. This process is illustrated in Figure 3.1. In what follows, SIMULATE REAL involves this operation.

Calculation of CF_{fuzzy}

Just as CF_{expert} is presented by the expert, we need a way to obtain CF_{fuzzy} from the fuzzy simulation. By doing this, we benefit from the comparison of the two rules in terms of their CF values. However, since the derivation of the CF_{expert} involves a subjective opinion, as well as a certain amount of uncertainty, there is no theoretical formulation to calculate the CF_{fuzzy} whose derivation process is exactly the same as that of the CF_{expert} . Our solution is to define an equation so its result agrees with a *human intuition* as much as possible. We used a *weighted average method* to create such intuition. Given a simplex rule, we define the CF_{fuzzy} by using the weighted average method:

$$CF_{fuzzy} = \frac{\sum_{j=1}^n (\mu_A(x_j) \times \mu_B(y_j))}{\sum_{j=1}^n \mu_A(x_j)}, \quad (3.1)$$

where

$x_j, j = 1, 2, \dots, n$, denote real values on X in the fuzzy set A , and

$y_j, j = 1, 2, \dots, n$, denote real values on Y obtained from SIMULATE REAL using x_j .

The validity of calculating CF_{fuzzy} using the weighted average method is shown as in Figure 3.2. The CF_{fuzzy} using Equation (3.1) is 1.0 and 0.0 for Figure 3.2 (a) and Figure 3.2 (b), respectively. The results match our intuition. When the CF falls into some range between the above two extreme cases (i.e., 0.0 and 1.0) as shown in Figure 3.3, we can intuitively say that each member in A supports the conclusion B with a higher confidence, the greater CF we get. Using (3.1), the CF_{fuzzy} for Figure 3.3 (a) is :

$$\begin{aligned} CF_{fuzzy} &= \frac{(0.5 \times 0.5) + (1.0 \times 1.0) + (0.5 \times 0.5)}{0.5 + 1.0 + 0.5} \\ &= 0.75, \end{aligned}$$

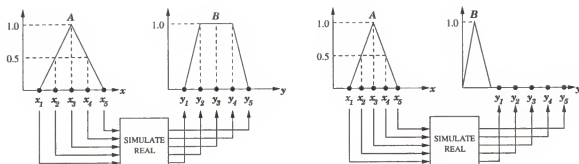


Figure 3.2. All members or none of members support the conclusion
 (a) All members of A support the conclusion B with full confidence; (b) None of the members of A support the conclusion B

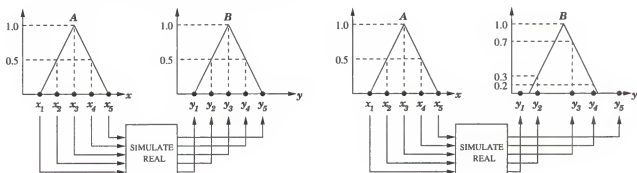


Figure 3.3. Some members support Conclusion
 (a) Members of A support the conclusion B with higher confidence compared to the case of (b); (b) Members of A support the conclusion B with less confidence compared to the case of (a)

and the CF_{fuzzy} for Figure 3.3 (b) is:

$$CF_{fuzzy} = \frac{(0.5 \times 0.3) + (1.0 \times 0.7) + (0.5 \times 0.2)}{0.5 + 1.0 + 0.5} = 0.48.$$

Example

Figure 3.4 illustrates how to perform fuzzy simulation using simplex rules, IF \mathcal{X} is X THEN \mathcal{Y} is B . The results of SIMULATE REAL are artificially made for the purpose of illustration. Applying Equation (3.1), we obtain CF_{fuzzy} by

$$CF_{fuzzy} = \frac{(1.0 \times 0.5) + (0.7 \times 0.5)}{0.3 + 0.7 + 1.0 + 0.7 + 0.3} = 0.21.$$

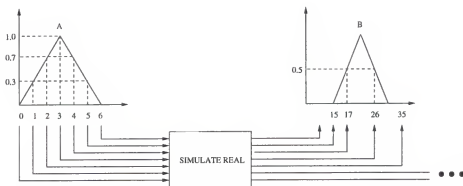


Figure 3.4. Fuzzy simulation using simplex rule

3.2.2 Fuzzy Simulation for Compound Rules with Arithmetic Operations

The *extension principle* [59] defined in *Definition 2.8* is a principle for fuzzifying crisp functions. It can be used to generalize crisp mathematical concepts into fuzzy sets. Owing to this principle, models and algorithms involving nonfuzzy variables can be extended to the case of fuzzy variables. By applying Equation (2.8) to a compound rule with arithmetic operation, an intermediate fuzzy set is obtained, and this set is used for fuzzy simulation. Consider a compound rule of the type

IF \mathcal{X} is $(A_1 \star A_2)$ THEN \mathcal{Y} is B ,

where \star is one of the four basic arithmetic operators (i.e., $+$, $-$, \times , \div).

Algorithm of Fuzzy Simulation

1. Apply Equation (2.8) to the rule premise.
2. Let Z be a resulting intermediate fuzzy set, and let a fuzzy simulation component such as a parameter p be defined as a fuzzy set Z , where

$$Z = \mu_Z(z_1)/z_1 + \mu_Z(z_2)/z_2 + \dots + \mu_Z(z_n)/z_n.$$

Assume the element of Z is identified by brackets (i.e., $Z[2] = z_2$).

3. For $j \in 1, 2, \dots, n$:

- (a) Let $\mathbf{p}[j] = Z[j]$.
- (b) SIMULATE REAL.
- (c) obtain $(\mu_B(y_j)/y_j)(t_e)$.

4. calculate CF_{fuzzy} .

Calculation of CF_{fuzzy}

Given a compound rule with arithmetic operations, we define the CF_{fuzzy} by using the weighted average method

$$CF_{fuzzy} = \frac{\sum_{j=1}^n (\mu_{A_1 \star A_2}(z_j) \times \mu_B(y_j))}{\sum_{j=1}^n \mu_{A_1 \star A_2}(z_j)}, \quad (3.2)$$

where

$z_j, j = 1, 2, \dots, n$, denote real values on a fuzzy set resulted from the arithmetic operation, $A_1 \star A_2$, and

$y_j, j = 1, 2, \dots, n$, denote real values on Y obtained from SIMULATE REAL using z_j .

Example

Let's assume that we want to perform fuzzy simulation using the following rule:

IF \mathcal{X} is $(A_1 + A_2)$ THEN \mathcal{Y} is B ,

where A_1 and A_2 are defined by Figure 3.5 (a) and Figure 3.5 (b).

By applying Equation (2.9) defined by

$$\mu_{A_1 + A_2}(z) = \max_{x=A_1 + A_2} (\mu_{A_1}(x) \wedge \mu_{A_2}(x)),$$

we obtain the following set of equation for the intermediate fuzzy set Z .

$$\mu_Z(1) = (0 \wedge 0.3) \vee (0 \wedge 0.1) = 0,$$

$$\mu_Z(2) = (0 \wedge 0.6) \vee (0.1 \wedge 0.3) \vee (0.3 \wedge 0) = 0.1,$$

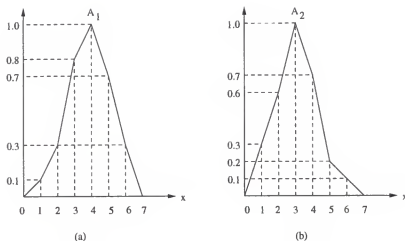


Figure 3.5. Two fuzzy sets for addition

$$\mu_Z(3) = (0 \wedge 1) \vee (0.1 \wedge 0.6) \vee (0.3 \wedge 0.3) \vee (0.8 \wedge 0) = 0.3,$$

$$\mu_Z(4) = (0 \wedge 0.7) \vee (0.1 \wedge 1) \vee (0.3 \wedge 0.6) \vee (0.8 \wedge 0.3) \vee (1 \wedge 0) = 0.3,$$

$$\mu_Z(5) = (0 \wedge 0.2) \vee (0.1 \wedge 0.7) \vee (0.3 \wedge 1) \vee (0.8 \wedge 0.6) \vee (1 \wedge 0.3) \vee (0.7 \wedge 0) = 0.6,$$

$$\begin{aligned} \mu_Z(6) = & (0 \wedge 0.1) \vee (0.1 \wedge 0.2) \vee (0.3 \wedge 0.7) \vee (0.8 \wedge 1) \vee (1 \wedge 0.6) \vee (0.7 \wedge 0.3) \vee \\ & (0.3 \wedge 0) = 0.8, \end{aligned}$$

$$\begin{aligned} \mu_Z(7) = & (0 \wedge 0) \vee (0.3 \wedge 0.3) \vee (0.7 \wedge 0.6) \vee (1 \wedge 1) \vee (0.8 \wedge 0.7) \vee (0.3 \wedge 0.2) \vee \\ & (0.1 \wedge 0.1) \vee (0 \wedge 0) = 1, \end{aligned}$$

$$\begin{aligned} \mu_Z(8) = & (0.1 \wedge 0) \vee (0.3 \wedge 0.1) \vee (0.8 \wedge 0.2) \vee (1 \wedge 0.7) \vee (0.7 \wedge 1) \vee (0.3 \wedge 0.6) \vee \\ & (0 \wedge 0.3) = 0.7, \end{aligned}$$

$$\mu_Z(9) = (0.3 \wedge 0) \vee (0.8 \wedge 0.1) \vee (1 \wedge 0.2) \vee (0.7 \wedge 0.7) \vee (0.3 \wedge 1) \vee (0 \wedge 0.6) = 0.7,$$

$$\mu_Z(10) = (0.8 \wedge 0) \vee (1 \wedge 0.1) \vee (0.7 \wedge 0.2) \vee (0.3 \wedge 0.7) \vee (0 \wedge 1) = 0.3,$$

$$\mu_Z(11) = (1 \wedge 0) \vee (0.7 \wedge 0.1) \vee (0.3 \wedge 0.2) \vee (0 \wedge 0.7) = 0.2,$$

$$\mu_Z(12) = (0.7 \wedge 0) \vee (0.3 \wedge 0.1) \vee (0 \wedge 0.2) = 0.1,$$

$$\mu_Z(13) = (0.3 \wedge 0) \vee (0 \wedge 0.1) = 0.$$

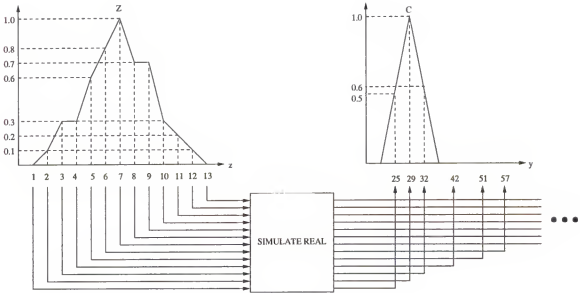


Figure 3.6. Fuzzy simulation of compound (addition) rule

Figure 3.6 shows the fuzzy set Z and the result of fuzzy simulation using Z . Again, the results of SIMULATE REAL was created arbitrarily for illustration purposes. Using Equation (3.2), we can calculate CF_{fuzzy} by

$$CF_{fuzzy} = \frac{(0.1 \times 1.0) + (0.3 \times 0.6)}{0.1 + 0.3 + 0.3 + 0.6 + 0.8 + 1.0 + 0.7 + 0.7 + 0.3 + 0.2 + 0.1} = 0.06.$$

3.2.3 Fuzzy Simulation for Compound Rules with Logic Operations

When the expert's rule premise involves logical operators such as *and* or *or*, the *rule of conjunctive composition* and *rule of disjunctive composition* [61] defined in Definition 2.10 is applied to obtain a possibility distribution. This distribution is used for fuzzy simulation. Consider a compound rule of the type

$$\text{IF } (\mathcal{X} \text{ is } A) * (\mathcal{Y} \text{ is } B) \text{ THEN } \mathcal{Z} \text{ is } C,$$

where $*$ denotes any logical operator.

Algorithm of Fuzzy Simulation

1. If $*$ is *and* operator,

then apply the *rule of conjunctive composition* to the rule premise and calculate a possibility distribution $\pi(x, y)$.

If $*$ is *or* operator,

then apply the *rule of disjunctive composition* to the rule premise and calculate a possibility distribution $\pi(x, y)$.

2. Let fuzzy simulation components such as \mathbf{p} and \mathbf{q} be defined as fuzzy sets A and B , respectively, where

$$A = \mu_A(x_1)/x_1 + \mu_A(x_2)/x_2 + \dots + \mu_A(x_n)/x_n,$$

$$B = \mu_B(y_1)/y_1 + \mu_B(y_2)/y_2 + \dots + \mu_B(y_n)/y_n.$$

Assume the elements of A and B are identified by brackets (i.e., $A[2] = x_2$ and $B[2] = y_2$).

3. For $i \in 1, 2, \dots, m$

For $j \in 1, 2, \dots, n$

Let $\mathbf{p}[i] = A[i]$.

Let $\mathbf{q}[j] = B[j]$.

SIMULATE REAL.

obtain $(\mu_C(z_{ij})/z_{ij})(t_e)$.

4. calculate CF_{fuzzy} ,

where m and n are the number of elements in A and B , respectively.

Notice that in the rule defined above, the universal set of the fuzzy variables A and B are not identical. Otherwise, instead of $\pi(x, y)$, we can get a more simplified fuzzy set as an intermediate set by *Definition 2.6* and *Definition 2.7* for disjunction and conjunction, respectively.

Calculation of CF_{fuzzy}

Given a compound rule with logic operations, CF_{fuzzy} is defined by using the weighted average method

$$CF_{fuzzy} = \frac{\sum_{i=1}^m \sum_{j=1}^n (\mu_{A*B}(x_i, y_j) \times \mu_C(z_{ij}))}{\sum_{i=1}^m \sum_{j=1}^n \mu_{A*B}(x_i, y_j)}, \quad (3.3)$$

where $*$ denotes logical operator.

Example

Let's assume that we want to perform fuzzy simulation using the following compound rule:

IF (\mathcal{X} is A) and (\mathcal{Y} is B) THEN \mathcal{Z} is C ,

where A and B are defined as

$$A = \text{small} = 1/1 + 0.6/2 + 0.1/3,$$

$$B = \text{large} = 0.1/1 + 0.6/2 + 1/3.$$

By applying the *rule of conjunctive composition*, the predicate (\mathcal{X} is A) and (\mathcal{Y} is B) yields the following possibility distribution:

$$\begin{aligned} \pi(x, y) &= \{[\mu_{A \text{ and } B}(x_1, y_1)/(x_1, y_1)], [\mu_{A \text{ and } B}(x_1, y_2)/(x_1, y_2)], \\ &= [\mu_{A \text{ and } B}(x_1, y_3)/(x_1, y_3)], [\mu_{A \text{ and } B}(x_2, y_1)/(x_2, y_1)], \\ &\quad \dots, [\mu_{A \text{ and } B}(x_3, y_3)/(x_3, y_3)]\} \\ &= \{[0.1/(1, 1)], [0.6/(1, 2)], [1/(1, 3)], [0.1/(2, 1)], [0.6/(2, 2)], [0.6/(2, 3)], \\ &\quad [0.1/(3, 1)], [0.1/(3, 2)], [0.1/(3, 3)]\}. \end{aligned}$$

Let's assume that we have the result as shown in Figure 3.7 after performing SIMULATE REAL on this $\pi(x, y)$. Using Equation (3.3), we can calculate CF_{fuzzy} by

$$CF_{fuzzy} = \frac{(1.0 \times 0.5) + (0.1 \times 0.8) + (0.6 \times 0.3)}{1.0 + 0.1 + 0.6} = 0.44.$$

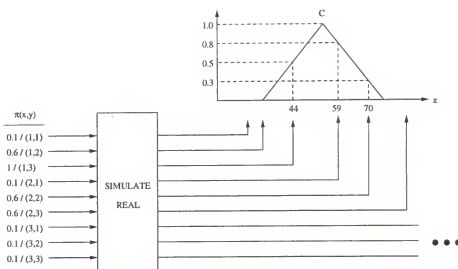


Figure 3.7. Fuzzy simulation of compound (conjunction) rule

In this chapter, we presented a new fuzzy simulation approach. We showed that how the fuzzy simulation handles *possiblistic uncertainty* by means of three approximate reasoning tools and the weighted average method. In the next chapter, we present an environment for isolating inconsistency between the expert rules and an assumed quantitative model. The fuzzy simulation approach forms a basis for building such an environment. This environment handles the *linguistic vagueness* discussed in Section 2.1.2 and supports interactive user control.

CHAPTER 4

A METHOD FOR ISOLATING INCONSISTENCY

The purpose of this chapter is to provide an interactive environment to check for consistency and resolve inconsistencies between the qualitative and the quantitative models. Based on the fuzzy simulation approach discussed in the previous chapter, this environment also handles the *linguistic vagueness* in the expert rules. The en-

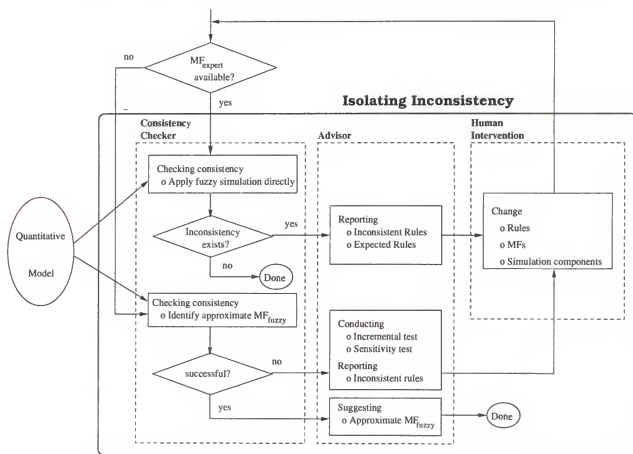


Figure 4.1. An environment for isolating inconsistency

vironment is shown at Figure 4.1. As shown in this figure, the method for isolating

inconsistency consists of three major components: **Consistency Checker**, **Advisor** and **Human Intervention**.

Consistency Checker performs two major tasks: checking consistency by applying the fuzzy simulation directly, and checking consistency by identifying approximate MFs . Whenever MF_{expert} is available (i.e., when the expert provides complete fuzzy sets as estimates for linguistic terms), it performs the first task by comparing the rule consequences from the fuzzy simulation against those from the expert in terms of CF . When the amount of inconsistency is out of range, all inconsistent rules and their expected rules from the fuzzy simulations are suggested through **Advisor**.

When there is no MF_{expert} *a priori* (i.e., when expert provides central point or interval estimates for linguistic terms), **Consistency Checker** performs consistency-checking by identifying an approximate MF_{fuzzy} . Using fuzzy simulation, it first focuses on discovering another important knowledge source - *linguistic definitions* in expert's rules. It tries to produce approximate definitions where the rules from fuzzy simulation maximally match against the rules from the expert. If such definitions can be generated with a fairly good match between the two rule sets, these linguistic definitions are suggested through **Advisor**, and consistency checking is finished by returning an answer, "*consistent*." Otherwise, the answer "*inconsistent*" is returned, and **Advisor** performs an *incremental test* or a *sensitivity test* to find out the source of inconsistency.

Using all informations from **Advisor**, we start to resolve the inconsistency. For this process, **Human Intervention** is permitted: either the expert rules (including CF_{expert} , MF_{expert}) or the simulation components can be modified interactively. Every time these modifications occur, **Consistency Checker** is reinvoked with visual aids, so that the user can easily recognize the effect of the modifications. Therefore, the overall process now involve *humans in the loop* during the process of checking

consistency and resolving inconsistency. We start with this chapter by introducing the quantitative measurements of inconsistency that we have employed.

4.1 Measurements of Inconsistency

The consistency between two types of models can be measured by the difference between CF presented by experts and CF calculated from fuzzy simulation on each rule. For each rule, we define its *Local Inconsistency*, LI , by

$$LI = |CF_{fuzzy} - CF_{expert}|. \quad (4.1)$$

Global inconsistency is measured by summing up such differences in the entire rule set. Thus, using the LI , we define the *Global Inconsistency*, GI , in a rule set by

$$GI = \sum_{i=1}^m LI_i, \quad (4.2)$$

where m = total number of rules.

Searching for the largest LI enables us to identify the most inconsistent rule (i.e., the worst case rule) between two different knowledge sources. Moreover, calculating the GI in this way allows us to measure the total amount of inconsistency. Note that given two rule sets and their GI , a *slightly* better GI does not always mean that the rule set leading to this GI is more consistent than the others. This results from *possiblistic uncertainty* of confidence factors derived by the expert [60] or *measurement uncertainty*, due to an inability of a measuring instrument to overcome its limiting finite resolution [27]. Using the Equation (4.2), we will say two given qualitative and quantitative models are *consistent* if

$$GI < \varepsilon, \quad (4.3)$$

where ε is a *consistency criterion* specified by user.

Any inconsistency found may be due to:

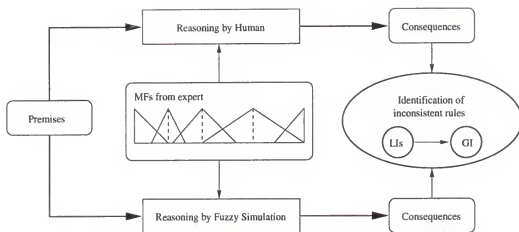


Figure 4.2. Identification process of inconsistent rules

- an inadequate conceptual model. Specifically, underlying assumptions, model structure, logic, mathematical relations and particular pieces of causal relationship may be inadequate,
- an improperly implemented conceptual model, or
- improperly designed expert rules. Specifically, improper rule structure, inadequate fuzzy value boundaries and central points or possibilistic uncertainty (confidence factor) may cause the inconsistency.

4.2 Checking Consistency When MF_{expert} is Available

By taking MF_{expert} as an input of fuzzy simulation as introduced in Chapter 3, we obtain each rule consequence associated with a CF_{fuzzy} . Using the CF_{expert} and CF_{fuzzy} pair, we get the LI in each rule by Equation (4.1). Then, the GI is obtained from Equation (4.2). This process is shown at Figure 4.2. The most inconsistent rule is considered the rule which has the largest LI . The GI can be used for a performance index. Thus, when any sources of inconsistent components are modified, a comparison between the current GI and the previous GI indicates whether this modification is a good decision or not. Once the inconsistent rules are

identified, we must be careful not to eliminate the possibility that the CF_{expert} was derived in different manner, and that it is not the same as the weighted average method defined in Equation (3.1), Equation (3.2) and Equation (3.3). For reducing such a possibility, **Advisor** generates rules for all possible consequences associated with the CF_{fuzzy} , whose values are calculated by the weighted average method.

4.3 Checking Consistency When MF_{expert} is Unavailable

The previous section showed how to check for consistency when linguistic terms are defined *a priori*. Checking for consistency was possible because fuzzy simulations used the expert's precisely predefined linguistic definitions. The algorithm presented here allows us to check for consistency with minimal information provided, such as *central point estimates*. In this section, we assume that, at the very least, the central point estimates for linguistic values are provided by the expert. For a complete discussion of the various estimate forms that can be covered in this study, see Section 4.3.3.

Checking for consistency is possible by generating a set of MF_{fuzzy} where $RULE_{expert}$ and $RULE_{fuzzy}$ maximally match. If such definitions (i.e., MF_{fuzzy}) are generated with a *GI* of less than *consistency criterion*, ε , $RULE_{expert}$ and $RULE_{fuzzy}$ are considered to be *consistent* with such definitions. However, if any set of MF_{fuzzy} does not lead to a *GI* of less than ε , then this means that we cannot find any linguistic definitions that properly fill the gap between two models. This implies that a discrepancy exists between $RULE_{expert}$ and $RULE_{fuzzy}$. In this case, **Advisor** performs either an *incremental test* or a *sensitivity test* to identify inconsistent rules.

4.3.1 Process in General

By forcing fuzzy simulation to produce $RULE_{fuzzy}$ and CF_{fuzzy} which are maximally close to $RULE_{expert}$ and CF_{expert} , respectively, we can discover a set of MF_{fuzzy}

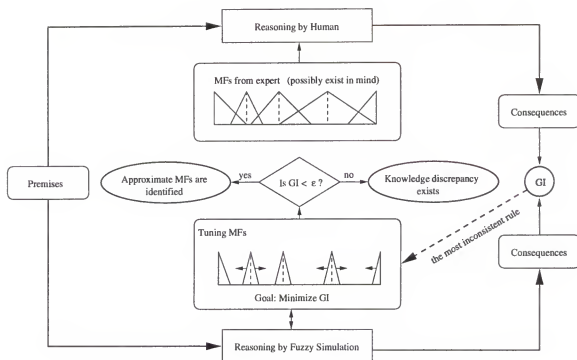


Figure 4.3. Identification process of approximate MF_{fuzzy}

between these two different models. This optimization process for generating the MF_{fuzzy} is shown in Figure 4.3. The purpose of this process is not to generate fine-tuned fuzzy membership functions, but to decide whether any set of approximate membership functions exist in which the two models lead to consistent conclusions. This process can be stated as “search for a set of MF_{fuzzy} which leads to a minimum GI for all rules, and then check if this GI is less than a given *consistency criterion*, ϵ .”

4.3.2 Heuristic Function and Search Method for Generating Approximate MF_{fuzzy}

To find such a set of MF_{fuzzy} , our algorithm uses the following heuristic function, goal, and search methods in its tuning process:

- heuristic function: GI .
- goal: minimization of GI .

- search method: gradient descent search for finding a minimum GI by always moving in the direction in which the decreasing rate of change is the greatest. Particularly, for each iteration,
 - pick the most inconsistent rule and then
 - pick such a subset of the MF_{fuzzy} in the rule that increasing the spreads of this subset by Δd reduces the GI to the greatest amount.

If Δd is too small, then fuzzy membership functions are adjusted very slowly. If a large Δd is chosen, the convergence may be faster, but optimal spreads may be missed. A possible solution is to pick Δd in an adaptive manner. That is, if the GI is decreasing rapidly, take big steps, but if the GI is decreasing slowly, take small steps.

Figure 4.4 shows that how the tuning algorithm uses this evaluation function and the search method to generate the MF_{fuzzy} . In this figure, a rule set is assumed to be composed of three rules, $R1$, $R2$ and $R3$. Our purpose in this figure is to find a set of approximate MF_{fuzzy} for all linguistic values that leads to the smallest GI . The tuning process begins with the minimal-size fuzzy sets centered on the central point estimates. Then it increases the size of appropriate fuzzy sets by Δd , where the appropriate fuzzy sets are selected using the above strategy. Let us assume that the first fuzzy simulation for each rule provides the information that $R2$ is the worst case rule. By selecting the $R2$, the algorithm encounters states, 2, 3, 4, ..., 8 which denote the every possible sets of MF_{fuzzy} after tuning a subset of the MF_{fuzzy} in $R2$. In this way, a *state* is defined as a set of MF_{fuzzy} . For example, if $R2$ contains three fuzzy sets A , B and C , the *state* 2 through *state* 8 denote each MF_{fuzzy} after tuning $\{A\}$, $\{B\}$, $\{C\}$, $\{A, B\}$, $\{B, C\}$, $\{A, C\}$ and $\{A, B, C\}$, respectively. Since the fuzzy simulation using *state* 4 leads to the lowest GI , *state* 4 (i.e., MF_{fuzzy} after $\{C\}$ is tuned) is selected for the next tuning process. In this way, more than zero fuzzy sets

are tuned at each iteration of the algorithm. When the algorithm reaches a point where such a modification does not reduce the GI compared to the previous GI , this sequence of tuning stops. However, this algorithm may reach either a *local minima*,

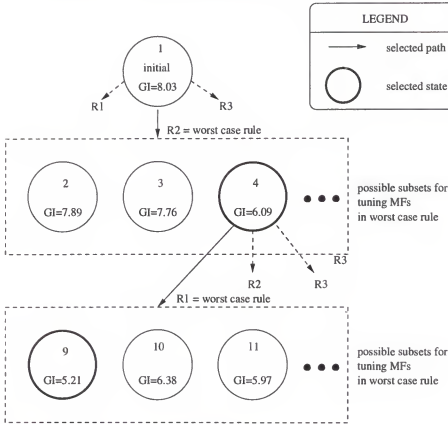


Figure 4.4. Heuristic function GI and search for the smallest GI

a *plateau*, or a *ridge*, because it keeps track of only the current states, and does not look ahead beyond the immediate neighbors of that state. Among many possible ways [20, 37, 39, 40] to deal with these problems, we adopt a *random-restart gradient descent search* [40] by conducting a series of gradient descent searches from randomly generated promising states, running each until it makes no discernible progress. For implementing this strategy, we employ a *promising state criterion*, θ defined in terms of GI , which is used for randomly selecting promising states. For example, promising

states can be defined as

$$\{\text{states} \mid GI < \theta\}, \quad (4.4)$$

where θ is specified by the user. To improve the search productivity, we always pick a state which doesn't lead to the same modification which has already been conducted. Putting these all together and using the following example of the expert's rules, we illustrate the random-restart gradient descent search in Figure 4.5.

R1: IF A THEN C

R2: IF B THEN D

R3: IF A and B THEN D

In this figure, the set representation in a node denotes one subset of MFs being tuned, and the numeric value in a node refers to a GI after this subset is tuned. Notice that *state 21* is rejected, even though it has the smallest GI at that point. Because this path leads to the same sequence of modification, $D \rightarrow A \rightarrow C \rightarrow B$, which already has been performed when we visited *state 11*, we reject *state 21*. Before presenting a detailed algorithm for identifying the approximate MF_{fuzzy} , we first introduce the expert's estimates that serve as a clue for locating initial positions of MF_{fuzzy} .

4.3.3 Various Forms of Expert's Estimates on Linguistic Terms

The expert's estimates can be one of the following various forms, depending on the uncertainty about his (or her) linguistic terms:

- central point estimates,
- interval estimates,
- approximate fuzzy membership functions, such as triangular or trapezoid fuzzy numbers and
- fuzzy membership functions with their complete definitions.

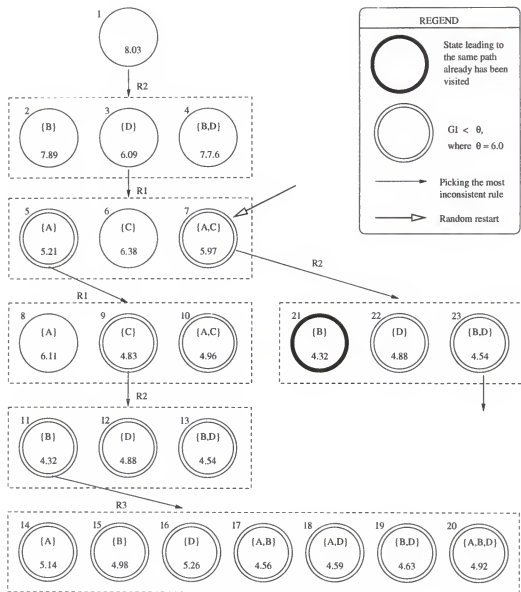


Figure 4.5. Random-restart gradient descent search

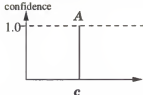
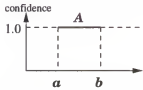
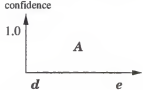
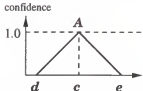
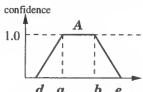
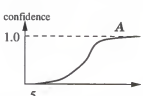
Type	Form of estimate	Explanation	Discretization into fuzzy space
1	central point estimate	When experts present the center point c of A	
2	Interval of full confidence	When experts present the interval $[a, b]$ of A with a full confidence	
3	Interval of min-max range	When experts present the min-max range $[d, e]$ of A	
4	Approximate fuzzy number	When experts present both the center point c and the min-max range $[d, e]$ of A	
		when experts present both the interval $[a, b]$ of A with a full confidence and its min-max range $[d, e]$	
5	Complete fuzzy number	When experts present the complete definition of A	

Figure 4.6. Five types of estimates about the linguistic terms

Figure 4.6 shows the five types of estimates and the way to discretize such an uncertainty into a fuzzy real space. From expert's point of view, the lower on the scale a type is located, the easier it is to estimate. Note that estimating *Type 3* is harder than *Type 1* and *Type 2*, since the exact extreme points of that interval are difficult to determine.

The algorithm for generating MF_{fuzzy} has been devised for handling the first two types of estimates (when *Type 4* or *Type 5* is presented, tuning MF_{fuzzy} itself is unnecessary, because the approximate or complete forms of linguistic values are already given). Even though the algorithm presented in the next section deals with the central point estimates, it can be easily extended to cover *Type 2* as well. That is, as shown in Figure 4.7, the meaning of the linguistic term of *Type 1* can be approximated by a symmetric, triangular fuzzy number. Likewise, the meaning of *Type 2* can be approximated by a symmetric, trapezoidal fuzzy number. The difference between those two types is only the shape of the fuzzy numbers. Therefore, these cases can be handled in the same way, if we represent every fuzzy numbers in our algorithm by 4-tuple symmetric fuzzy numbers $[d, a, b, e]$ as shown in Figure 4.7. For example, the membership degree of two types in this figure can be derived using the same equations defined by

$$\mu(x) = \begin{cases} 0 & \text{if } x \leq d \\ 1 - \frac{2(a-x)}{e-d-(b-a)} & \text{if } d < x < a \\ 1 & \text{if } a \leq x \leq b \\ 1 - \frac{2(x-b)}{e-d-(b-a)} & \text{if } b < x < e \\ 0 & \text{if } x \geq e. \end{cases}$$

4.3.4 Algorithm to Generate MF_{fuzzy}

As shown in Figure 4.8, the algorithm to generate the approximate MF_{fuzzy} consists of six steps. All steps are explored in detail.

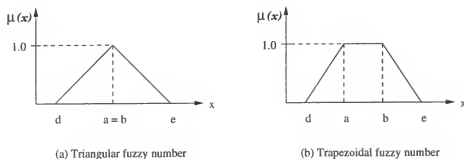
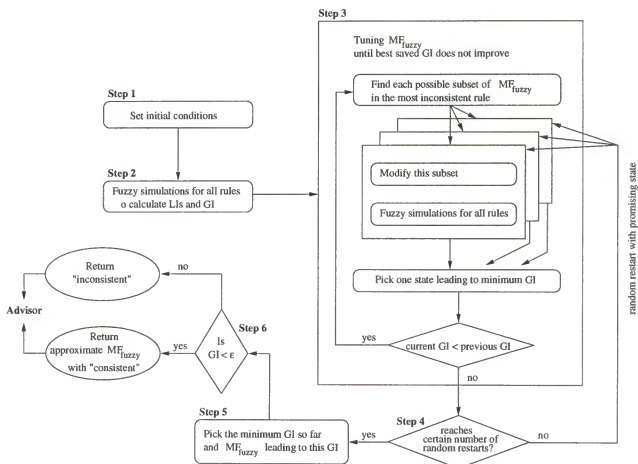


Figure 4.7. A 4-Tuple Fuzzy Number

(a) Triangular fuzzy number; (b) Trapezoidal fuzzy number

Figure 4.8. An algorithm for MF_{fuzzy} generation

Step 1: Set Initial Conditions

The following items are initially set by the user.

- a tuning size, Δd , of MF_{fuzzy} ,
- a *consistency criterion*, ε ,
- a *promising state criterion*, θ and
- number of random restarts to be conducted.

Initial sizes of MF_{fuzzy} can be determined such that each range of MF_{fuzzy} is $2\Delta d$, based on a central point estimate. The user can initially set the last three items to certain numbers, but it is better determine this later, because the characteristics of these values can be observed after a few iterations of the algorithm.

Step 2: Fuzzy Simulations for All Rules

For each $RULE_{expert}$, a fuzzy simulation is executed as discussed in Chapter 3. Sampling is first done at the central point, then in both directions by increasing and decreasing Δd until right before these points exceed the width of the $MF_{premise}$. The LI for each rule and the GI for a whole-rule set are calculated by Equation (4.1) and Equation (4.2), respectively.

Step 3: Tuning MF_{fuzzy} until best saved GI does not improve

The purpose of this step is to incrementally reduce a GI by selecting the most inconsistent rule and modifying a proper subset of MF_{fuzzy} among the all subsets of MF_{fuzzy} in the most inconsistent rule until we eventually reach the smallest GI . A series of tuning processes stops at this point. During this step, we maintain two lists: a $Visit_{list}$ and a $Random_{list}$. The $Visit_{list}$ is a list to save a state which already has been *visited*. This list is used to avoid revisiting the same states. The $Random_{list}$ is a list which saves a state whose GI is less than given *promising state criterion*, θ .

This list is used when we *randomly* pick a promising state for random-restarts. As shown in Figure 4.8, this step consists of four substeps.

- **Step 3.1:** Find each possible subset of MF_{fuzzy} in the most inconsistent rule. That is, first, by picking a rule which has the largest LI , we find the most inconsistent rule. Next, all possible subsets of the MF_{fuzzy} in this rule are obtained from the power set of the MF_{fuzzy} . Then, for each subset, we execute **Step 3.2**.
- **Step 3.2:** For each subset of MF_{fuzzy} , we increase the elements in each subset by Δd and execute fuzzy simulations for all rules to calculate each LI .
- **Step 3.3:** Pick one state leading to a minimum GI . This state is likely to be the best candidate for reducing GI as a whole. Then, update the $Visit_{list}$ and $Random_{list}$ by adding this state to the $Visit_{list}$, and adding the other states whose GIs are less than θ to the $Random_{list}$.
- **Step 3.4:** Compare this GI to the previous GI . The purpose of this substep is to make sure that the best candidate obtained from **Step 3.3** actually improves the situation. Thus, a *stop condition* for the tuning process can be written as $Current\ GI \geq previous\ GI$? If the condition is satisfied, our algorithm proceeds to **Step 4**. Otherwise, the current GI is saved as a previous GI , and the entire **Step 3** is executed until the stop condition is satisfied.

Step 4: Random-restarts

This step continues to execute **Step 3** with random-restarts, each time reinitializing GI to a maximum value, until the algorithm reaches a user specified number of restarts. Note that

- before executing **Step 3.1**, we randomly choose a state in the $Random_{list}$ and immediately deleted it from the list and

- in **Step 3.3**, the state selected should not be in the $Visit_{list}$.

Step 5: Pick a State Leading to a Minimum GI

Among GI s resulting from a series of random-restarts, pick a minimum GI and its MF_{fuzzy} .

Step 6: Return Results Depending on the Consistency Criterion, ε

If the GI above is less than ε , then it means that the algorithm was capable of generating the MF_{fuzzy} in which two models exist in a consistent manner. In this case, the MF_{fuzzy} is returned as approximate linguistic definitions for both the $RULE_{expert}$ and the $RULE_{fuzzy}$. Otherwise, the algorithm returns “*inconsistent*.” However, we may still find a source of inconsistency by performing the *incremental test* or the *sensitivity test* discussed in the following section.

4.3.5 Identify Inconsistent Rules

Even though any set of MF_{fuzzy} could not be found with a GI of less than ε in the previous section, we may still identify inconsistent rules which are responsible for the failure by using two methods. The first method is the *incremental test*, and the second is the *sensitivity test*. **Advisor** performs either or both tests depending on the user’s request. In both tests, one should not assume that a newly identified inconsistent rule is the only rule which needs to be analyzed. All existing rules which may cause conflicts with this rule should also be analyzed to resolve the inconsistency. Note that the tests presented here do not work well when a significant portion of a model is inconsistent with the model to which we want to compare it.

Incremental Test

In this test, we can identify inconsistent rules by observing the rate of changes of the GI as we add rules incrementally and run the MF_{fuzzy} -generation algorithm repeatedly. When the rate of change of the GI is significantly increased by adding $rule_k$, the $rule_k$ can be considered as a more inconsistent rule than previously added

rules. Once this $rule_k$ is added, observing the rate of change of GI is meaningless, since from this point, the inconsistent component (i.e., $rule_k$) also contributes to tuning the MF_{fuzzy} into an incorrect direction, thereby making the GI less credible. Thus, before the test, an arrangement should be made so that more reliable rules be employed on the test before less reliable rules. For this reason, the incremental test is order-dependent and requires more heuristics than the sensitivity test discussed in the following section. However, when one or more domain experts are available, we can apply a *fuzzy individual or group preference ordering method* [27] for obtaining the ordering within a given compatibility. Having such information available, the advantage of the incremental test over the sensitivity test is that it requires fewer computation demands. The algorithm for the incremental test can be described as:

1. Let n be the total number of rules for the incremental test. Assume that rules are ordered from the most reliable rule, $rule_1$, to the least reliable rule, $rule_n$.
2. Divide these rules into two group, R and U , depending on their reliability such that group R contains the reliable rules, $rule_1$ to $rule_j$, and group U contains the unreliable rules, $rule_{j+1}$ to $rule_n$.
3. Let T be a set of rules for incremental test. Initially T is empty.
4. For $i = 1$ to j
 - (a) Add $rule_i$ to T .
 - (b) Run MF_{fuzzy} -generation algorithm on T .
 - (c) Observe the rate of change of GI .
 - (d) If the rate is significant,

then report $rule_i$ as inconsistent, and stop.
5. For $i = j + 1$ to n

- (a) Add $rule_i$ to T .
- (b) Run MF_{fuzzy} -generation algorithm on T .
- (c) Observe the rate of change of GI .
- (d) If the rate is significant,
 - then report $rule_i$ be inconsistent, and stop.
- (e) Delete $rule_i$ from T .

Note that for preventing human experts' possible errors, we inserted **Step 4(c)** and **Step 4(d)**, even though the rules being added here belong to the reliable group, R .

Sensitivity Test

A sensitivity test is a validation technique that can be used for both expert systems and simulations [35, 21, 29]. The general idea is to change the system input (i.e., values of variables or parameters) systematically over some range of interest, and study it by observing the effect. In our method, we can employ this idea to identify inconsistent rules by varying the participating rules. The sensitivity test begins with all rules except the first rule and executes the algorithm to generate MF_{fuzzy} . Then, all rules except the second rule are used to generate the MF_{fuzzy} . In this way, given the n rules, the sensitivity test involves a total of n executions of MF_{fuzzy} -generation algorithm, each time with $n - 1$ rules. When we detect a significant improvement of the GI during this process, we can consider the rule that did not participate at this step of the test as an inconsistent rule, in comparison to other rules already joined.

The advantage of this method over the incremental test is that the arrangement of rules based on their reliability is not necessary - thereby eliminating the need for the experts' opinion. However, the computation burden is more severe than the incremental test, since each execution of MF_{fuzzy} -generation algorithm involves $n - 1$ rules. The general algorithm for sensitivity test can be written as:

1. Let n be a total number of rules for sensitivity test.
2. Let T be a set of rules for sensitivity test. Initially T has entire rules.
3. For $i = 1$ to n
 - (a) Delete $rule_i$ from T .
 - (b) Run MF_{fuzzy} -generation algorithm on T .
 - (c) If a significant improvement of GI is detected,
then report $rule_i$ be inconsistent.
 - (d) Insert $rule_i$ to T .

4.4 Time Complexity

For the analysis of the time complexities of the fuzzy simulation method and MF_{fuzzy} generation algorithm, we consider the following factors:

- number of rules: n ,
- number of simplex rules: s ,
- number of sampling points of a fuzzy value: p ,
- number of input fuzzy variables in compound rules: m ,
- number of possible subsets for modifying MFs in the most inconsistent rule: b ,
- depth in a search tree: d and
- number of random-restarts: r .

4.4.1 Time Complexity for Fuzzy Simulation

For a simplex rule, the total p elements of a fuzzy set in the rule premise are issued to each independent fuzzy simulation. Therefore, the time complexity for executing

fuzzy simulations for s simplex rules is

$$O(sp). \quad (4.5)$$

For a compound rule, an intermediate fuzzy set Z is first obtained from the rule premise, and a fuzzy simulation is executed for each element in Z . This process involves a *cartesian product* over $m - 1$ variables (i.e., except for the variable in the rule consequence) in the rule. This leads to the time complexity $O(p^m)$. Therefore, the time complexity for executing fuzzy simulations for all compound rules is

$$O((n - s)p^m). \quad (4.6)$$

Combining complexity (4.5) and (4.6), the total time complexity for executing fuzzy simulations for all rules is

$$O(np^m). \quad (4.7)$$

4.4.2 Time Complexity for MF_{fuzzy} Generation

Using the factors, r and d defined above, we can rewrite the overall algorithm for the MF_{fuzzy} generation in Section 4.3.4 as follows:

1. set initial MFs
2. fuzzy simulations for all rules
3. pick the most inconsistent rule and set current GI
4. for $i = 1$ to r

while current GI \neq previous GI

(a) for $j = 1$ to b

i. modify this subset

ii. execute fuzzy simulations for all rules

(b) pick one state leading to minimum GI

5. pick best saved GI

In the algorithm skeleton above, **Step 4(a)** determines the branch factor of the search tree. In the worst case, the branch factor is $2^m - 1$ by ignoring ϕ . Therefore, using (4.7), the time complexity involving **Step 4(a)** to **Step 4(b)** is

$$O(n2^m p^m). \quad (4.8)$$

The depth d in an arbitrary search tree is determined by the condition (5). Thus, the overall time complexity for generating the MF_{fuzzy} is

$$O(rdn2^m p^m). \quad (4.9)$$

The overall time complexity shown above demonstrates that, in a worst case, the number of fuzzy variables that can appear in any compound rule (i.e., m) dominates the overall time complexity. Besides, as we pointed out in Section 2.2.3, m also affects the number of fuzzy rules, n , as well [10]. However, most fuzzy applications so far have had few variables and have been in control [28]. This makes the time complexity shown in (4.9) manageable. In other words, the running time of MF_{fuzzy} generation that we encounter in practical situations are mostly tractable problems.

In this chapter, we described an interactive environment for isolating inconsistent knowledge between the expert rules and the quantitative model. To handle the linguistic vagueness in the expert rules, two separate procedures (i.e., applying fuzzy simulation directly and generating MF_{fuzzy}) were presented depending on the types of estimates in the expert rules. In the next two chapters, we illustrate the applications of the presented methodology.

CHAPTER 5 FULTON: STEAMSHIP MODELING

For a practical application of the method discussed in the previous chapters, we will consider FULTON, a model of a steam-powered ship, as shown in Figure 5.1 [30]. When the fuel valve is open, fuel flows and the furnace heats the sea water in the boiler assembly; when the fuel valve is closed, no fuel flows and the furnace stops heating the water. Heating the sea water produces steam, which is gathered and goes to the turbine, making the steamboat movable. The remaining steam is condensed into liquid in the condenser and is pumped back to the boiler.

Among the four components in Figure 5.1, let's assume that we are interested in the boiler assembly, particularly the relation between temperature (T) of the sea water and the amount of steam (A_s) gathered in the boiler assembly. In the following two sections, we discuss two models which represent this knowledge quantitatively and qualitatively. Then, we apply our method to isolate any inconsistency between these two models.

5.1 Quantitative Model of Boiler Assembly

Consider the boiler assembly in Figure 5.1. The fuel valve is determined to be in one of two states: open or close. Then, depending on the valve position, the behavior of the boiler assembly can be represented by four states of FSA as in Figure 5.2. The low level continuous models for $M1, \dots, M4$ in that figure are defined as shown below by combining Newton's law with the capacitance law [18].

1. (M1) COLD: $T = \alpha$, $\dot{A}_w = \dot{A}_{in}$, $\dot{A}_s = 0$,

2. (M2) HEATING: $\dot{T} = k_1(100 - T)$, $\dot{A}_w = -k_2 + \dot{A}_{in}$, $\dot{A}_s = \dot{T} * k_3$,

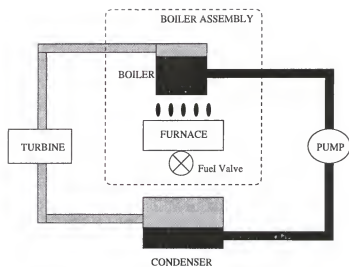


Figure 5.1. a model of a steam-powered ship

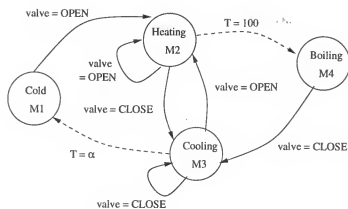


Figure 5.2. Four state automaton controller for the boiler assembly

3. (M3) COOLING: $\dot{T} = k_4(\alpha - T)$, $\dot{A}_w = -k_5 + \dot{A}_{in}$, $\dot{A}_s = \dot{T} * k_6$ and

4. (M4) BOILING: $T = 100$, $\dot{A}_w = -k_7 + \dot{A}_{in}$, $\dot{A}_s = k_8$,

where

$k_i, i = 1, \dots, 8$ are rate constants,

α = the ambient temperature of the water,

T = the temperature of sea water,

A_w = the amount of sea water,

A_s = the amount of steam gathered in the boiler, and

A_{in} = the amount of water increased in the boiler by pumping water from the pump assembly.

5.2 Qualitative Model of Boiler Assembly

In the expert's point of view, one of the easiest ways to model the physical behaviors of the boiler assembly is to represent that knowledge into natural language. Since the expert is interested in the relationship between the temperature of the sea water and the amount of steam, he or she can form an associative rule-based model by mapping a single input (the amount of time the fuel valve is open) into a single output (the amount of steam) as shown in Figure 5.3. However, as we can see

Valve Open Time	Amount of Steam	CF _{expert}
very_very_short	very_very_little	0.9
very_short	very_little	0.6
short	little	0.6
slightly_moderate	little	0.6
moderate	medium	0.6
slightly_long	slightly_much	0.9
long	slightly_much	0.5
very_long	much	0.9

Figure 5.3. Simplex rules for the boiler assembly

in Figure 5.4 and Figure 5.5, the rate of change in the temperature when the fuel

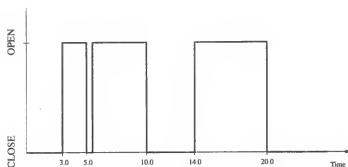


Figure 5.4. Fuel valve input

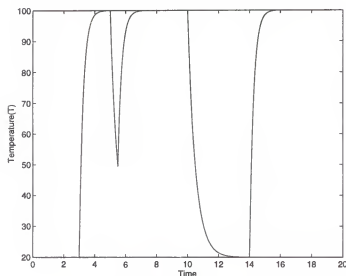


Figure 5.5. Water temperature

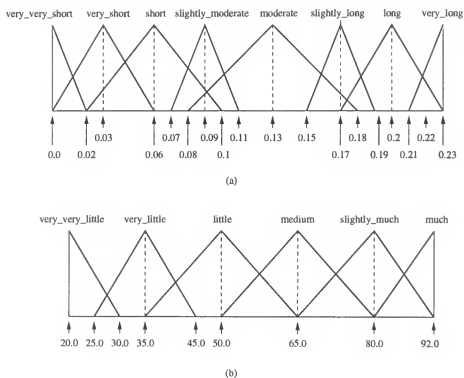
valve is open is different from the rate of change when the valve is closed. Taking this observation into account, the expert can construct a more complex knowledge base using compound rules to measure the time-dependent dynamic behavior of the system. Figure 5.6 illustrates this process. Putting Figure 5.3 and Figure 5.6 together and using the expert's definition of linguistic terms defined by Figure 5.7, we obtained a complete rule-based model from the expert for the boiler assembly.

5.3 Checking Consistency When MF_{expert} is Available

Once the complete rule-based model is obtained from the expert, the logical step is to run fuzzy simulations on this model, and get $RULE_{fuzzy}$ and CF_{fuzzy} to see

Valve Open Time	Valve Close Time	Amount of Steam	CF _{expert}
long	very_short	much	0.5
slightly_moderate	very_long	medium	0.8
moderate	short	slightly_much	0.5
short	long	little	0.5
very_long	very_very_short	much	0.7
slightly_long	slightly_moderate	much	0.6
very_short	slightly_long	little	0.9
very_very_short	moderate	very_very_little	0.9

Figure 5.6. Compound rules for the boiler assembly

Figure 5.7. Definitions of the linguistic terms for the boiler assembly
(a) $MF_{premise}$ in the expert's rule; (b) MF_{conseq} in the expert's rule

if any inconsistency exists. This path is represented as bold arrows in Figure 5.8. Inconsistencies are identified in terms of *LIs* and *GI* between these two rule sets. We assume that the *consistency criterion*, ε , is set to 0.5 by the user. Figure 5.9

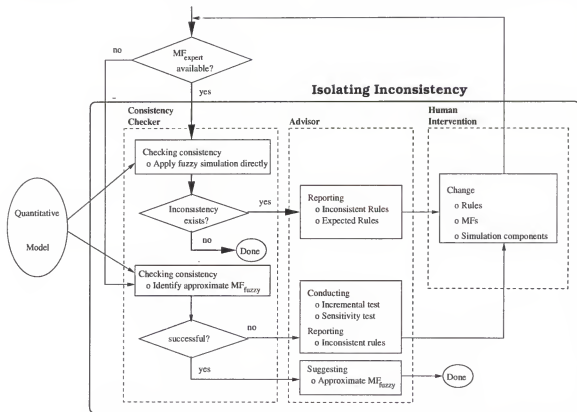


Figure 5.8. Checking consistency when MF_{expert} is available

shows the result of fuzzy simulations. This figure shows that *15th* rule is the most inconsistent, and its *LI* greatly affects the overall *GI*. To alleviate the human's resolving efforts, **Advisor** generates the expected rules from the fuzzy simulation by computing the CF_{fuzzy} for all possible consequences, given the premise part of the *15th* rule. Figure 5.10 shows this result. If the user agrees with **Advisor** by deciding to replace the original *rule*₁₅ with the *2nd* rule in Figure 5.10, he or she gets a fairly small *GI* of 0.34, as shown in Figure 5.11. Since this *GI* is less than *consistency criterion*, 0.5, we conclude that the two models are now consistent.

Valve Open Time	Valve Close Time	Amount of Steam	CF _{expert}	CF _{fuzzy}	LI
very_very_short	N/A	very_very_little	0.9	0.87	0.03
very_short	N/A	very_little	0.6	0.59	0.01
short	N/A	little	0.6	0.58	0.02
slightly_moderate	N/A	little	0.6	0.60	0.00
moderate	N/A	medium	0.6	0.64	0.04
slightly_long	N/A	slightly_much	0.9	0.88	0.02
long	N/A	slightly_much	0.5	0.50	0.00
very_long	N/A	much	0.9	0.94	0.04
long	very_short	much	0.5	0.53	0.03
slightly_moderate	very_long	medium	0.8	0.76	0.04
moderate	short	slightly_much	0.5	0.49	0.01
short	long	little	0.5	0.51	0.01
very_long	very_very_short	much	0.7	0.71	0.01
slightly_long	slightly_moderate	much	0.6	0.61	0.01
very_short	slightly_long	little	0.9	0.26	0.64
very_very_short	moderate	very_very_little	0.9	0.90	0.00
GI = 0.93					

Figure 5.9. The result of fuzzy simulation for the boiler assembly

Valve Open Time	Valve Close Time	Amount of Steam	CF _{fuzzy}
very_short	slightly_long	very_very_little	0.04
very_short	slightly_long	very_little	0.46
very_short	slightly_long	little	0.26
very_short	slightly_long	medium	0.00
very_short	slightly_long	slightly_much	0.00
very_short	slightly_long	much	0.00

Figure 5.10. Rules generated from Advisor for the boiler assembly

Valve Open Time	Valve Close Time	Amount of Steam	CF _{expert}	CF _{fuzzy}	LI
very_very_short	N/A	very_very_little	0.9	0.87	0.03
very_short	N/A	very_little	0.6	0.59	0.01
short	N/A	little	0.6	0.58	0.02
slightly_moderate	N/A	little	0.6	0.60	0.00
moderate	N/A	medium	0.6	0.64	0.04
slightly_long	N/A	slightly_much	0.9	0.88	0.02
long	N/A	slightly_much	0.5	0.50	0.00
very_long	N/A	much	0.9	0.94	0.04
long	very_short	much	0.5	0.53	0.03
slightly_moderate	very_long	medium	0.8	0.76	0.04
moderate	short	slightly_much	0.5	0.49	0.01
short	long	little	0.5	0.51	0.01
very_long	very_very_short	much	0.7	0.71	0.01
slightly_long	slightly_moderate	much	0.6	0.61	0.01
very_short	slightly_long	little	0.5	0.46	0.04
very_very_short	moderate	very_very_little	0.9	0.90	0.00
GI = 0.34					

Figure 5.11. Two consistent models

5.4 Checking Consistency When MF_{expert} is Unavailable

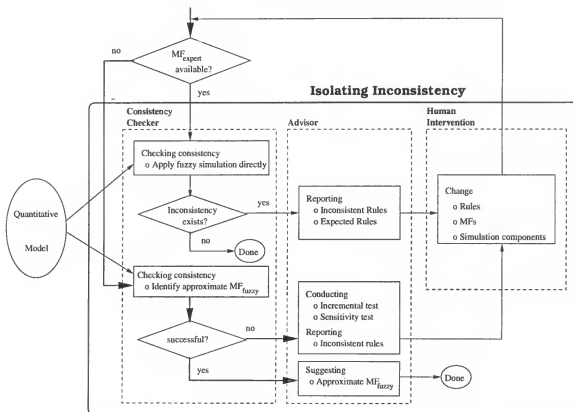


Figure 5.12. Checking consistency when MF_{expert} is unavailable

The path that is going to be explored in this section is represented as bold arrows in Figure 5.12. Since two different outcomes (i.e., *consistent* or *inconsistent*, depending on whether any approximate set of MF_{fuzzy} can be generated with a *GI* less than the *consistency criterion* or not) are produced exclusively through the consistency checking procedure, we illustrate each case in the following two sections.

5.4.1 The Case Where Approximate MF_{fuzzy} is Successfully Generated

The previous section showed that rules defined in Figure 5.11 are consistent. Therefore, if we use the rules and the central point estimates defined in Figure 5.13, then we can illustrate that the approximate set of MF_{fuzzy} can be found with a *GI* less than *consistency criterion*, 0.5. We illustrate this process step by step.

$MF_{premise}$	central point	MF_{conseq}	central point
very_very_short	0.0	very_very_little	20.0
very_short	0.03	very_little	35.0
short	0.06	little	50.0
slightly_moderate	0.09	medium	65.0
moderate	0.13	slightly_much	80.0
slightly_long	0.17	much	92.0
long	0.2		
very_long	0.23		

Figure 5.13. Central point estimates for $MF_{premise}$ and MF_{conseq}

Set Initial Conditions

Figure 5.14 shows the initial MF_{fuzzy} . The initial size is set to 0.02 for each $MF_{premise}$, and 4.0 for MF_{conseq} . To generate MF_{fuzzy} , the tuning size, Δd , is set to 0.005 for $MF_{premise}$, and 2.0 for MF_{conseq} . For random-restarts, the *promising state criterion*, θ , is set to 5.0. After all tuning processes are done, if a best saved *GI* is less than *consistency criterion* of 0.5, then we conclude that we found an approximate set of MF_{fuzzy} successfully.

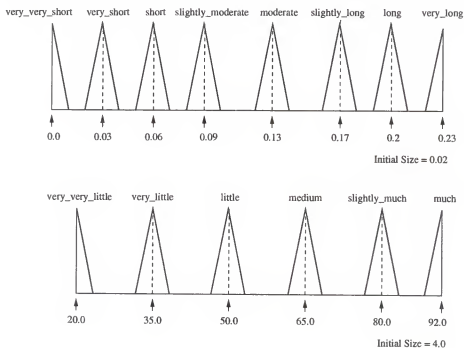


Figure 5.14. Initial MF_{fuzzy}

Fuzzy Simulations for All Rules

The result of fuzzy simulations using the initial MF_{fuzzy} is shown in Figure 5.15, where $rule_{10}$ is identified as the most inconsistent rule.

Valve Open Time	Valve Close Time	Amount of Steam	CF_{expert}	CF_{fuzzy}	LI
very_very_short	N/A	very_very_little	0.9	1.00	0.10
very_short	N/A	very_little	0.6	0.50	0.10
short	N/A	little	0.6	0.00	0.60
slightly_moderate	N/A	little	0.6	0.00	0.60
moderate	N/A	medium	0.6	0.00	0.60
slightly_long	N/A	slightly_much	0.9	0.50	0.40
long	N/A	slightly_much	0.5	0.00	0.50
very_long	N/A	much	0.9	1.00	0.10
long	very_short	much	0.5	0.00	0.50
slightly_moderate	very_long	medium	0.8	0.00	0.80
moderate	short	slightly_much	0.5	0.00	0.50
short	long	little	0.5	0.00	0.50
very_long	very_very_short	much	0.7	1.00	0.30
slightly_long	slightly_moderate	much	0.6	0.00	0.60
very_short	slightly_long	very_little	0.5	0.00	0.50
very_very_short	moderate	very_very_little	0.9	1.00	0.10
GI = 6.80					

Figure 5.15. The result of fuzzy simulations using the initial MF_{fuzzy}

Tuning MF_{fuzzy} Until Best Saved GI Does Not Improve

All possible subsets of MF_{fuzzy} in $rule_{10}$ are

{slightly_moderate}, {very_long}, {medium}, {slightly_moderate, very_long},
 {slightly_moderate, medium}, {very_long, medium} and {slightly_moderate,
 very_long, medium}.

After executing fuzzy simulations for each state, the smallest GI of 6.3 was found by modifying a subset {slightly_moderate, medium} as shown in Figure 5.16 and Figure 5.17. Since this GI is less than the previous GI , 6.8, the current GI , 6.3, is saved as the best GI , and **Step 3** in Section 4.3.4 is repeated until the GI does not improve. Figure 5.18 shows the actual GUI when the stop condition was met after a series of tuning processes. At this point, **Step 4** (random-restarts) begins,

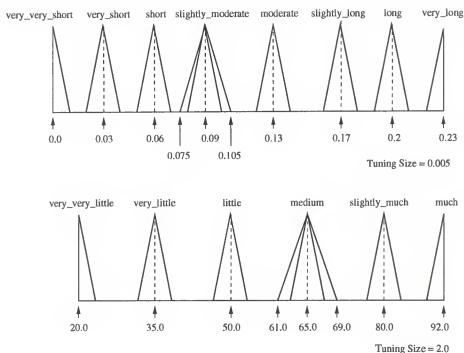


Figure 5.16. Increasing the size of *slightly_moderate* and *medium* by Δd

Valve Open Time	Valve Close Time	Amount of Steam	CF _{expert}	CF _{fuzzy}	LI
very_very_short	N/A	very_very_little	0.9	1.00	0.10
very_short	N/A	very_little	0.6	0.50	0.10
short	N/A	little	0.6	0.00	0.60
slightly_moderate	N/A	little	0.6	0.00	0.60
moderate	N/A	medium	0.6	0.25	0.35
slightly_long	N/A	slightly_much	0.9	0.50	0.40
long	N/A	slightly_much	0.5	0.00	0.50
very_long	N/A	much	0.9	1.00	0.10
long	very_short	much	0.5	0.00	0.50
slightly_moderate	very_long	medium	0.8	0.25	0.50
moderate	short	slightly_much	0.5	0.00	0.50
short	long	little	0.5	0.00	0.50
very_long	very_very_short	much	0.7	1.00	0.30
slightly_long	slightly_moderate	much	0.6	0.00	0.60
very_short	slightly_long	very_little	0.5	0.00	0.50
very_very_short	moderate	very_very_little	0.9	1.00	0.10

GI = 6.30

Figure 5.17. The result of fuzzy simulations after *slightly_moderate* and *medium* are tuned

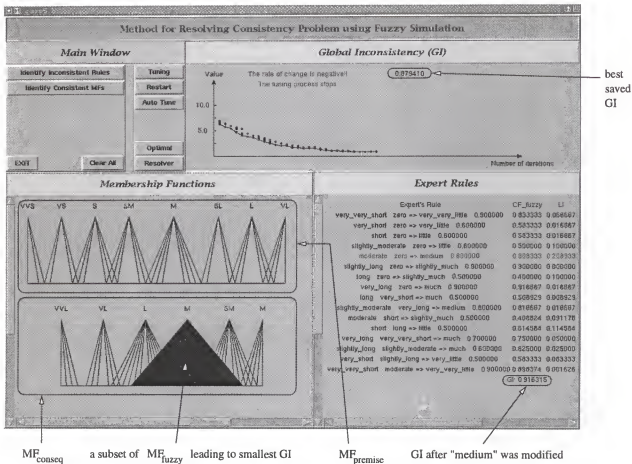
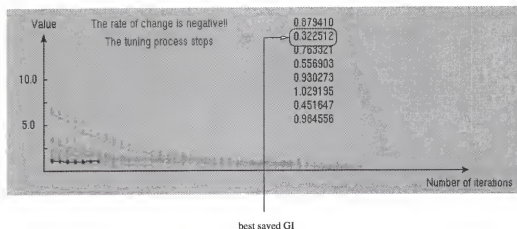
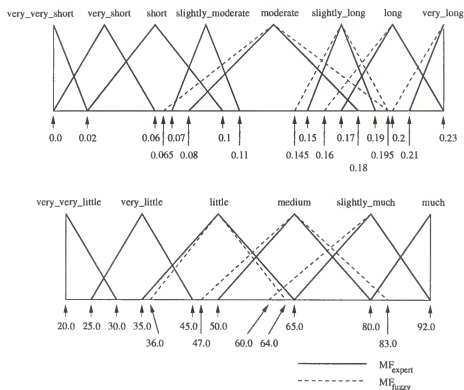


Figure 5.18. Satisfying stop condition

and Step 3 is reinvoked each time with a promising state whose GI is less than θ , 5.0. Figure 5.19 shows the trend of GI s after seven random-restarts. As we can see in this figure, the first and sixth random-restarts led to the GI of 0.32 and 0.45, respectively, which are less than the consistency criterion, 0.5. Thus, two sets of the MF_{fuzzy} leading to these GI s can be considered as good approximation of linguistic definitions, and we conclude that the expert's rules and the quantitative model are consistent if the two sets of MF_{fuzzy} are used. Figure 5.20 shows the comparison between the approximate set of MF_{fuzzy} which led to the smallest GI of 0.32, and the MF_{expert} defined in Fig 5.7.

Figure 5.19. Trend of GI after seven random-restartsFigure 5.20. The comparison between MF_{fuzzy} and MF_{expert}

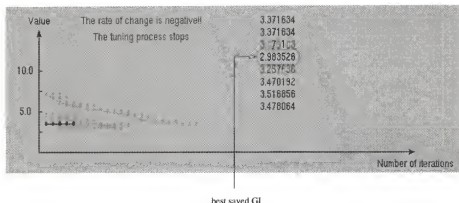


Figure 5.21. Trend of GI after seven random-restarts

5.4.2 The Case Where MF_{fuzzy} is Unsuccessfully Generated

Section 5.3 showed that expert's rules in Figure 5.9 are inconsistent because of $rule_{15}$. If we use these rules, we can artificially make the case where MF_{fuzzy} cannot be successfully generated. By executing the MF_{fuzzy} generation algorithm with seven random-restarts, we obtained the best saved GI of 2.98, as shown in Figure 5.21. In this figure, no random-restart lead to a GI less than the given consistency criterion, 0.5. To identify the causes of the inconsistency, we performed the incremental test and the sensitivity test discussed in Section 4.3.5.

Incremental Test

We assume that compound rules are more prone to inconsistency than simplex rules, because the consequences of all compound rules in the FULTON example require a more complex reasoning process involving two time-related variables. For this reason, we grouped the simplex rules, defined in Figure 5.3, into the reliable group, R , and the complex rules defined in Figure 5.6 into the unreliable group, U .

The result of the incremental test is shown at Figure 5.22. Each subfigure shows the result of executing the MF_{fuzzy} generation algorithm (i.e., Step 5(b) of the incremental test procedure in Section 4.3.5) with seven random-restarts right after we added each compound rule to the results obtained from incremental test using

the eight simplex rules. By observing the rapid change of GI in Figure 5.22 (g), we can regard the $rule_{15}$ as the catalyst that caused the inconsistency.

Sensitivity Test

Figure 5.23 shows the result of the sensitivity test on the rules defined in Figure 5.3 and Figure 5.6. We performed seven random-restarts for each test; each circle in this figure represents the GI we have obtained from each random-restart. A rule i in the x axis represents the rule which did not join for the MF_{fuzzy} generation at that particular test. As shown in this figure, execution of the MF_{fuzzy} generation algorithm (i.e., **Step 3(b)** of the sensitivity test procedure in Section 4.3.5) without $rule_{15}$ resulted in a significant improvement of GI , 0.42, compared to the other cases. This indicates that $rule_{15}$ is the catalyst of the inconsistency.

5.4.3 Human Intervention

In Figure 5.24, the path involving human intervention is represented as bold arrows. In Section 5.3, $rule_{15}$ was identified as an inconsistent rule when the rules defined in Figure 5.3 and Figure 5.6, and the linguistic definitions defined in Figure 5.7, were used. Even in a situation where such complete linguistic definitions were not available, the previous section showed that we were still able to identify $rule_{15}$ as a catalyst for the inconsistency by conducting the incremental test or the sensitivity test. With this information, the user may change rules (in this case, $rule_{15}$ or other rules which may be causing the conflict with $rule_{15}$), central point estimates, the definitions of linguistic terms in these rules or even simulation model components. We built a *GUI* as shown in Figure 5.25 to interactively visualize the system's responses according to these kinds of user's resolving trials. In this figure, the set of membership functions shown in **membership function editor** is the best set which led to the minimum GI of 2.98 in the previous section. However, as mentioned, this case fell into the category of unsuccessfully generated MF_{fuzzy} . Now,

using **membership function editor** and **rule editor**, the user can freely change these membership functions, as well as any rules including the *CFs*. Each time the user makes a modification, this updated input is issued to the fuzzy simulation, and new consistency checking results are provided in terms of *LI* and *GI* through **evaluation button**. In this chapter, we considered FULTON for a practical application of the presented methodology. All algorithms or procedures in Chapter 4 were demonstrated completely. For another application, we consider predator-prey population in the next chapter. In the chapter, we focus on the comparison between the model outputs of the expert rules and the outputs of the quantitative models before and after resolving inconsistency.

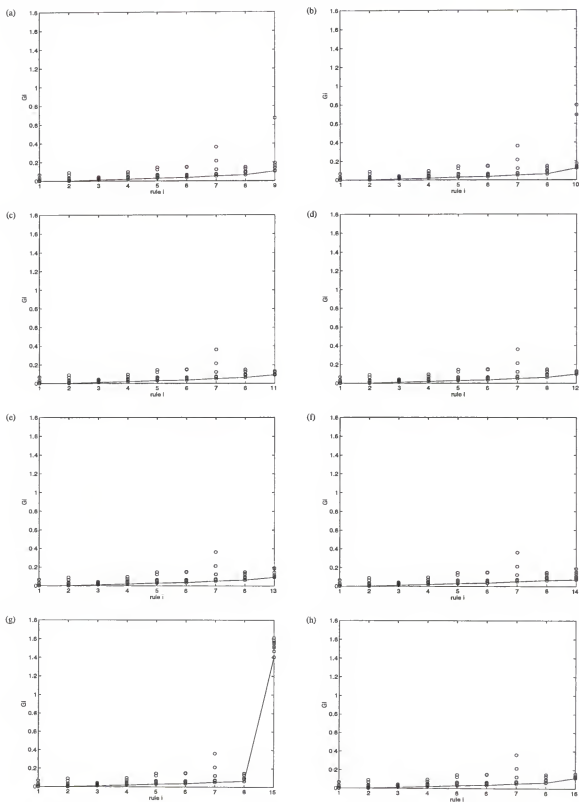


Figure 5.22. Incremental test for the boiler assembly

(a) adding *rule*₉; (b) adding *rule*₁₀; (c) adding *rule*₁₁; (d) adding *rule*₁₂;
 (e) adding *rule*₁₃; (f) adding *rule*₁₄; (g) adding *rule*₁₅; (h) adding *rule*₁₆

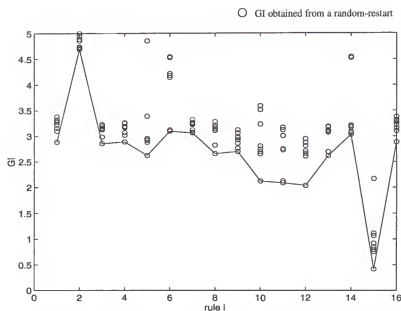


Figure 5.23. Sensitivity test for the boiler assembly

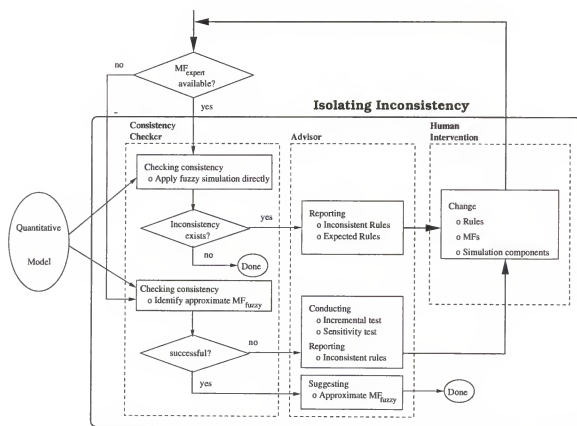


Figure 5.24. Human intervention

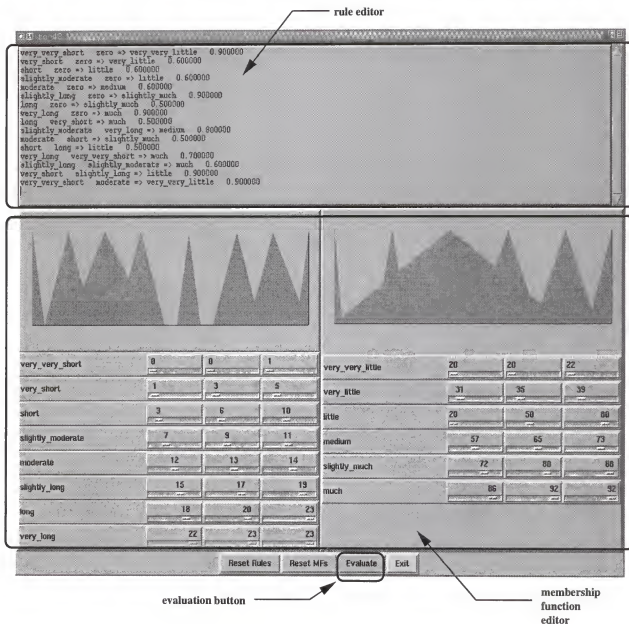


Figure 5.25. A GUI for human intervention

CHAPTER 6 PREDATOR-PREY POPULATION

For another illustration of our method, we considered a predator-prey population. Predator-prey models address the dynamic interaction between a predator species and its prey species. In this chapter, we consider two species: one single predator species \mathcal{P} and its single resource \mathcal{R} .

6.1 Qualitative Model

In general, there are two kinds of rules which explain the population dynamics in the predator-prey interaction: *migration rules* and *birth/death rules* [56, 55]. The migration rules describe the movement of individual \mathcal{P} and \mathcal{R} at any interval of time from one location to another. For example, a probability that \mathcal{P} will stay in a particular location depends on the configuration of \mathcal{R} and \mathcal{P} around that location. The birth/death rates of \mathcal{P} and \mathcal{R} depend on many factors, such as mortality rate, internal feeding state, predator's encounter rate with prey and so on. These factors are not independent of one another. For example, the mortality rate of \mathcal{P} depends on their internal feeding state, and the internal feeding state again depends on the encounter rate with \mathcal{R} .

Predator-prey dynamics also depends on the natural environment of \mathcal{P} and \mathcal{R} . When ecologists study the distribution of organisms, they try to discover the physical and biological factors that influence the presence or absence of particular species. In this section, we assume that there are two environmental factors affecting the distribution of \mathcal{P} and \mathcal{R} : scale and temperature of the region they live. Given two different environmental factors, an ecologist may predict the population of \mathcal{R} in a

short-term period by considering the causal relation of predator-prey interaction: “In a wide region, the possibility of \mathcal{P} ’s encounter rate with \mathcal{R} is relatively small, but if the temperature of the territory is warm, then other living food in that region may satisfy the \mathcal{P} (which makes the mortality rate low). Therefore, this keeps the population of \mathcal{R} from growing too much. On the contrary, in a cold region, the mortality rate of \mathcal{P} increases, which makes the population of \mathcal{R} crowded.” Using the above description, a knowledge engineer can come up with the following two rules.

IF \mathcal{P} ’s encounter rate with \mathcal{R} is *rare* and \mathcal{P} ’s mortality rate is *low*, THEN
the density of the \mathcal{R} is *slightly_crowded* (CF = 0.9),

IF \mathcal{P} ’s encounter rate with \mathcal{R} is *rare* and \mathcal{P} ’s mortality rate is *high*, THEN
the density of the \mathcal{R} is *crowded* (CF = 1.0).

After the knowledge acquisition process with the expert in this way, suppose that we obtained 25 rules and membership functions as shown in Figure 6.1 and Figure 6.2, respectively for explaining the population of prey \mathcal{P} based on the different combination of the environmental factors. With two inputs and five linguistic values for each of these, there are $5^2 = 25$ possible rules. In these rules, both triangular and trapezoidal membership functions are used.

6.2 Quantitative Model

To describe the dynamics involving growth and decline of the predator-prey population, differential or difference equations are often used. Such mathematical models are designed either for predictive purposes to make accurate short-term forecasts, or to identify generic characteristics and underlying principles. As one of the mathematical models, we consider the Lotka-Volterra predator-prey model [55]:

$$\begin{aligned}\frac{dx(t)}{dt} &= r x(t) \left(1 - \frac{x(t)}{K}\right) - \frac{a x(t) y(t)}{1 + a t_h x(t)}, \\ \frac{dy(t)}{dt} &= \frac{a c x(t) y(t)}{1 + a t_h x(t)} - e y(t),\end{aligned}$$

Rule	Encounter rate of predator with prey	Mortality rate of predator	Density of prey	CF
1	rare	very_low	slightly_crowded	0.90
2	rare	low	crowded	1.00
3	rare	moderate	crowded	1.00
4	rare	slightly_high	crowded	1.00
5	rare	high	crowded	1.00
6	slightly_rare	very_low	crowded	0.40
7	slightly_rare	low	crowded	0.50
8	slightly_rare	moderate	crowded	0.90
9	slightly_rare	slightly_high	crowded	1.00
10	slightly_rare	high	crowded	1.00
11	medium	very_low	scarce	0.60
12	medium	low	nominal	0.90
13	medium	moderate	crowded	0.50
14	medium	slightly_high	crowded	0.90
15	medium	high	crowded	1.00
16	slightly_frequent	very_low	scarce	0.80
17	slightly_frequent	low	scarce	0.80
18	slightly_frequent	moderate	slightly_scarce	0.30
19	slightly_frequent	slightly_high	slightly_crowded	0.30
20	slightly_frequent	high	crowded	1.00
21	frequent	very_low	scarce	0.80
22	frequent	low	scarce	0.60
23	frequent	moderate	slightly_scarce	0.30
24	frequent	slightly_high	nominal	0.10
25	frequent	high	crowded	0.80

Figure 6.1. Expert rules for the predator-prey population

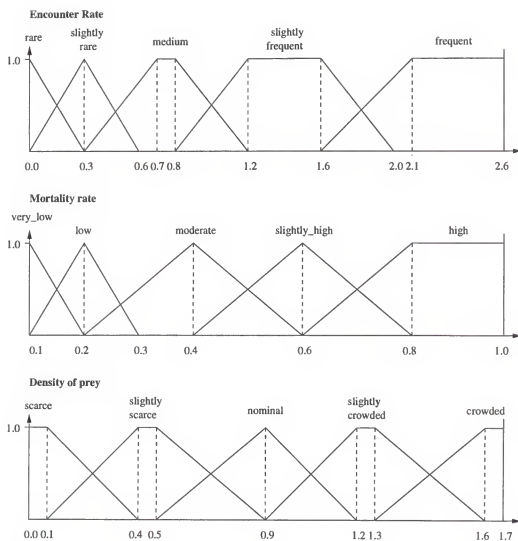


Figure 6.2. Fuzzy membership functions for the expert rules in Figure 6.1

where

$x(t)$, $y(t)$ = population densities of \mathcal{R} and \mathcal{P} as functions of time t , respectively,

r = \mathcal{R} population's intrinsic rate of increase,

K = \mathcal{R} 's *carrying capacity*, that is the population density the prey population would reach at an equilibrium in the absence of predation,

a = \mathcal{P} 's *encounter rate* with \mathcal{R} ,

c = a *conversion rate* which maps the \mathcal{P} consumption into the \mathcal{R} birth rate,

e = \mathcal{P} 's *mortality rate*, and

t_h = \mathcal{P} 's *handling time*.

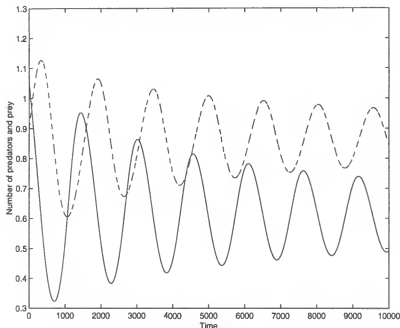


Figure 6.3. A time series graph for the predator-prey population

For example, using initial conditions $x(t) = 1.05$, $y(t) = 0.9$, $r = 1.2$, $K = 1.7$, $a = 1.9$, $e = 0.48$, $c = 0.9$ and $t_h = 1.0$, we obtain a time series graph for the predator-prey population as shown in Fig 6.3, in which the state variables (i.e., density of \mathcal{P} and

\mathcal{R}) are graphed against time. To solve the $x(t)$ and $y(t)$, we applied *Euler's method* [18].

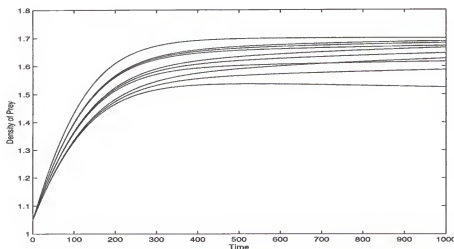


Figure 6.4. The result of fuzzy simulation on IF *rare* and *low* THEN *crowded*

6.3 Consistency Checking

We applied fuzzy simulation to the expert rules in Figure 6.1 using the membership functions defined in Figure 6.2 and initial conditions $x(t) = 1.05$, $y(t) = 0.9$, $r = 1.2$, $K = 1.7$, $c = 0.9$ and $t_h = 1.0$ of the Lotka-Volterra model. For example, Figure 6.4 shows the result of the fuzzy simulation on the expert's second rule, defined in Figure 6.1. To obtain this result, we first applied the *rule of conjunctive composition* to calculate the possibility distribution π of *rare* and *low*. Then each element of the possibility distribution is used for input of the fuzzy simulation, and this gives us a result in terms of membership degrees associated with *crowded*.

Finally, using the *weighted average method* introduced in Chapter 3, we obtain a CF_{fuzzy} , 1.0. In this way, all 25 expert rules are applied to the fuzzy simulations, and we obtained the results in Figure 6.5. In this figure, *rule*₁ is identified as the most inconsistent rule and the *rule*₁₂ as the second worst case rule. Figure 6.6 shows the suggestion from **Advisor**. By replacing the original *1st* and *12th* expert rules

Rule	Encounter rate of predator with prey	Mortality rate of predator	Density of prey	CF _{expert}	CF _{fuzzy}	LI
1	rare	very_low	slightly_crowded	0.90	0.05	0.85
2	rare	low	crowded	1.00	1.00	0.00
3	rare	moderate	crowded	1.00	1.00	0.00
4	rare	slightly_high	crowded	1.00	1.00	0.00
5	rare	high	crowded	1.00	1.00	0.00
6	slightly_rare	very_low	crowded	0.40	0.37	0.03
7	slightly_rare	low	crowded	0.50	0.46	0.04
8	slightly_rare	moderate	crowded	0.90	0.94	0.04
9	slightly_rare	slightly_high	crowded	1.00	1.00	0.00
10	slightly_rare	high	crowded	1.00	1.00	0.00
11	medium	very_low	scarce	0.60	0.60	0.00
12	medium	low	nominal	0.90	0.09	0.81
13	medium	moderate	crowded	0.50	0.46	0.04
14	medium	slightly_high	crowded	0.90	0.88	0.02
15	medium	high	crowded	1.00	1.00	0.00
16	slightly_frequent	very_low	scarce	0.80	0.78	0.02
17	slightly_frequent	low	scarce	0.80	0.79	0.01
18	slightly_frequent	moderate	slightly_scarce	0.30	0.29	0.01
19	slightly_frequent	slightly_high	slightly_crowded	0.30	0.31	0.01
20	slightly_frequent	high	crowded	1.00	0.98	0.02
21	frequent	very_low	scarce	0.80	0.77	0.03
22	frequent	low	scarce	0.60	0.64	0.04
23	frequent	moderate	slightly_scarce	0.30	0.32	0.02
24	frequent	slightly_high	nominal	0.10	0.15	0.05
25	frequent	high	crowded	0.80	0.80	0.00
GI = 2.06						

Figure 6.5. The result of fuzzy simulation for the predator-prey model

with their counterparts from fuzzy simulations, we achieved a *GI* of 0.38 as shown in Figure 6.7.

An idea of how the old and the new rule sets approximate the population of \mathcal{R} differently can be obtained by comparing the *response surfaces* of these two with one from Lotka-Volterra algebraic formula. The response surface of \mathcal{R} 's density for all 270 combinations of encounter rate = 0.1, 0.2, ..., 2.6 and mortality rate = 0.1, 0.2, ..., 1.0 from the Lotka-Volterra model is shown at Figure 6.8. To create the response surfaces from the fuzzy rule sets defined in Figure 6.1 and Figure 6.7, we used fuzzy logic (discussed in Section 2.2.3) with the *min-max composition* for inference

Rule	Encounter rate of predator with prey	Mortality rate of predator	Density of prey	CF _{fuzzy}
1	rare	very_low	scarce	0.00
	rare	very_low	slightly_scarce	0.00
	rare	very_low	nominal	0.00
	rare	very_low	slightly_crowded	0.05
	rare	very_low	crowded	0.95
12	medium	low	scarce	0.24
	medium	low	slightly_scarce	0.45
	medium	low	nominal	0.09
	medium	low	slightly_crowded	0.05
	medium	low	crowded	0.21

Figure 6.6. Rules suggested from **Advisor** for the predator-prey model

Rule	Encounter rate of predator with prey	Mortality rate of predator	Density of prey	CF _{expert}	CF _{fuzzy}	LI
1	rare	very_low	crowded	0.90	0.90	0.00
2	rare	low	crowded	1.00	1.00	0.00
3	rare	moderate	crowded	1.00	1.00	0.00
4	rare	slightly_high	crowded	1.00	1.00	0.00
5	rare	high	crowded	1.00	1.00	0.00
6	slightly_rare	very_low	crowded	0.40	0.37	0.03
7	slightly_rare	low	crowded	0.50	0.46	0.04
8	slightly_rare	moderate	crowded	0.90	0.94	0.04
9	slightly_rare	slightly_high	crowded	1.00	1.00	0.00
10	slightly_rare	high	crowded	1.00	1.00	0.00
11	medium	very_low	scarce	0.60	0.60	0.00
12	medium	low	slightly_scarce	0.40	0.40	0.00
13	medium	moderate	crowded	0.50	0.46	0.04
14	medium	slightly_high	crowded	0.90	0.88	0.02
15	medium	high	crowded	1.00	1.00	0.00
16	slightly_frequent	very_low	scarce	0.80	0.78	0.02
17	slightly_frequent	low	scarce	0.80	0.79	0.01
18	slightly_frequent	moderate	slightly_scarce	0.30	0.29	0.01
19	slightly_frequent	slightly_high	slightly_crowded	0.30	0.31	0.01
20	slightly_frequent	high	crowded	1.00	0.98	0.02
21	frequent	very_low	scarce	0.80	0.77	0.03
22	frequent	low	scarce	0.60	0.64	0.04
23	frequent	moderate	slightly_scarce	0.30	0.32	0.02
24	frequent	slightly_high	nominal	0.10	0.15	0.05
25	frequent	high	crowded	0.80	0.80	0.00

GI = 0.50

Figure 6.7. Two consistent models for the predator-prey model

and the *centroid* method for defuzzification. First, in the *min inference*, each *CF* was considered to calculate the membership degree of the consequence membership function. That is, given an encounter rate a and mortality rate e and a $Rule_j$, the truth value Tv_j of the output membership function D_j can be calculated by

$$Tv_j = (\mu_{E_j}(a) \wedge \mu_{M_j}(e)) \times CF_j, \quad (6.1)$$

where E_j and M_j are two fuzzy values defined in the premise of the $rule_j$, and \wedge is the minimum operator.

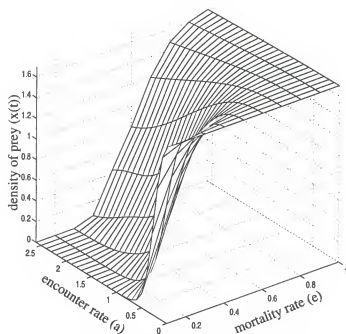


Figure 6.8. Population of prey obtained from the Lotka-Volterra model

Through the *min-inference*, the output membership function D_j is cut off at a height corresponding to the Tv_j . Then, we obtain a combined output membership function D' through the *max composition* by taking the pointwise maximum over all of the fuzzy sets obtained from the *min-inference* for all $Rule_j, j = 1, \dots, 25$. Finally,

we obtain the density d by the *defuzzification* formula,

$$d = \frac{\sum_{i=1}^n d_i \mu_{D'}(d_i)}{\sum_{i=1}^n \mu_{D'}(d_i)}, \quad (6.2)$$

where $d_i, i = 1, \dots, n$ is an element defined in the fuzzy set D' .

Figure 6.9 and Figure 6.10 shows the population of \mathcal{R} obtained in this way for the rule sets defined in Figure 6.1 and Figure 6.7, respectively. As we can see in these figures, by replacing the original *1st* and *12th* rules with the rules suggested from the fuzzy simulation, we can obtain a better approximation for the population of \mathcal{R} .

An explanation may be possible as to why the two models show inconsistency meshexpert1.epsmeabout the particular two causal relations defined in the *1st* rule and *12th* rule. First, when \mathcal{P} 's encounter rate with \mathcal{R} is *rare*, the mortality rate of the \mathcal{P} almost doesn't affect the density of the \mathcal{R} (by observing the *1st* rule to the *5th* rule in Figure 6.7). The interval between the two boundaries of *rare* defined by the expert (Figure 6.2) seems so narrow, that every combinations of *rare* and any value of the mortality rate result in the approximately same population of \mathcal{R} . Second, by observing the *11th* rule to the *13th* rule in that figure, when \mathcal{P} 's encounter rate with \mathcal{R} is *medium*, the increase of \mathcal{P} 's mortality rate from *very_low* to *low* results in a lesser change of the \mathcal{R} 's population than when it increases from *low* to *moderate*.

6.4 MF_{fuzzy} Generation

In this section, we illustrate how to obtain an approximate set of MF_{fuzzy} using the rules, defined in Figure 6.7, with only central points or intervals of full confidence, shown at Figure 6.2. For the MF_{fuzzy} -generation process, we set the tuning size $\Delta d = 0.1$ for the encounter rate, 0.05 for the mortality rate and 0.1 for the density of \mathcal{R} . The *promising state criterion*, θ , and the *consistency criterion*, ε , are set to 2.4 and 0.7, respectively. Under these settings, we conducted a total of 15 random-restarts during the MF_{fuzzy} -generation process and obtained the trend of GI , as shown at

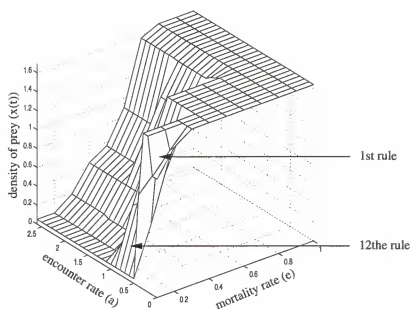


Figure 6.9. Population of prey obtained from the original expert rules defined in Figure 6.1

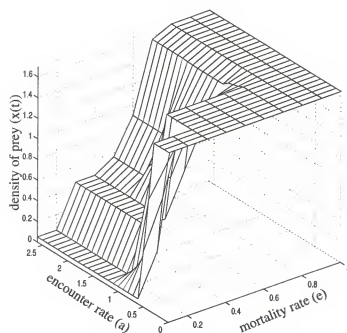


Figure 6.10. Population of prey obtained from the consistent rules defined in Figure 6.7

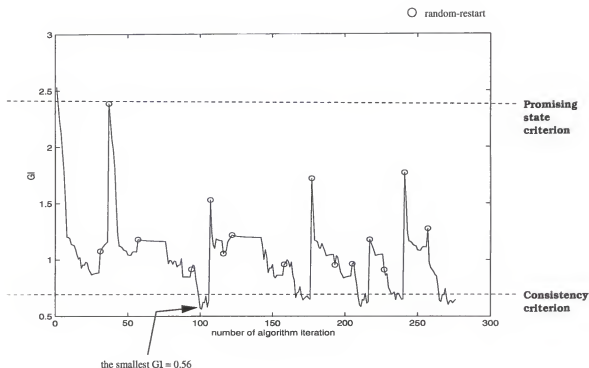


Figure 6.11. Trend of GI during 15 random-restarts

Figure 6.11. As we can see in that figure, a total of 36 sets of MF_{fuzzy} led to a GI of less than the *consistency criterion*, 0.7. From this, we conclude that the expert rules defined in Figure 6.7 and the Lotka-Volterra model are consistent with any of these MF_{fuzzy} . Figure 6.12 shows the comparison between the MF_{fuzzy} leading to the smallest GI of 0.56 against the original MF_{expert} defined in Figure 6.2. The response surface generated from this set of MF_{fuzzy} and the consistent rules is shown at Figure 6.13.

In this chapter, we considered another example, predator-prey population to illustrate the application of the presented methodology. Given the expert rules and the quantitative model, we demonstrated that how we made these two models more consistent by employing the methodology to isolate inconsistency and resolve inconsistency.

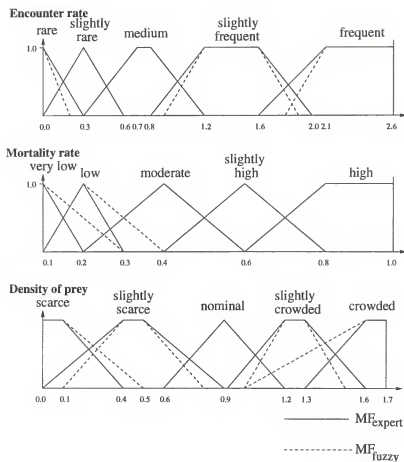


Figure 6.12. The comparison between MF_{fuzzy} and MF_{expert}

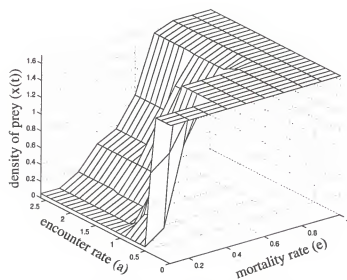


Figure 6.13. Population of prey obtained from MF_{fuzzy}

CHAPTER 7 FUTURE WORK

In this chapter, we first mention the limitations of the presented methodology and suggest ways improving or reducing such limitations in the future. Then, we suggest two promising application areas: application in control industries and application in MOOSE.

7.1 Limitations and Improvements

7.1.1 Resolving Inconsistency

In this study, the main concern is to isolate the particular piece of causal relations which show inconsistency between the expert's rules and an assumed quantitative model. However, after isolating these inconsistent rules, most efforts to resolve the problem rely on **Human Intervention**. We can add the following capabilities to **Advisor** to support the human efforts to resolve the inconsistencies:

- suggesting the ideal rule structure by determining the optimal number of fuzzy variables and values,
- locating the expert's missing rules if they exist and
- locating the expert rules upon which we should elaborate.

7.1.2 Performance Index

In this research, Global Inconsistency (GI) serves as a performance index to search for better linguistic definitions where expert rules and quantitative model match maximally. Since the value of GI depends on the number of rules, this value may differ from an application to an application. To make this value independent on

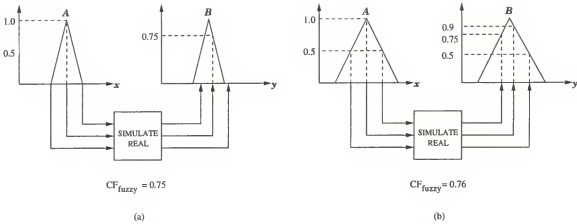


Figure 7.1. Local optimality using the weighted average method

(a) Fuzzy simulation with the initial size of fuzzy sets A and B; (b) Fuzzy simulation with the desirable size of fuzzy sets A and B

any particular application, a normalizing procedure (such as dividing by the number of rules) is necessary. After such a standard representation of GI is obtained, the goodness of GI can be more strictly studied in the future.

7.1.3 Local Optimality During MF_{fuzzy} Generation

The *weighted average method* employed in this study may cause local optimality during MF_{fuzzy} generation when stand-alone rules exist in expert rules. Here, we define a *stand-alone rule* as a rule whose fuzzy values are never used in other rules. To illustrate this problem, we consider a stand-alone expert rule,

IF \mathcal{X} is A THEN \mathcal{Y} is B (0.7).

Suppose that we obtain CF_{fuzzy} of 0.75 by executing the fuzzy simulation with the initial size of fuzzy sets A and B, as shown in Figure 7.1 (a). This is certainly a problem if the membership function defined in Figure 7.1 (b) is the goal for which we are looking. The stand-alone rule will never be picked as the worst case rule by the MF_{fuzzy} generation algorithm because $CF_{expert} \approx CF_{fuzzy}$ at its initial sizes of A and B. We can consider the following approaches to reduce such local optimality problems:

- Always allow the best case rule to be picked at the end of the tuning process (i.e., **Step 3** of the algorithm described in Section 4.3.4). Continue this process to see if the *GI* eventually decreases and reaches its previous minimal point again.
- Detect all stand-alone rules and report them to the user. If the user is able to provide more rules using the fuzzy sets A and B, then the rule is not a stand-alone rule any more.

7.2 Application

7.2.1 Application in Control Industries

As one of the promising applications of the presented methodology, we can consider the application involving Proportional Integral Derivative (PID) controller. To illustrate this kind of applications, we particularly consider *subway control system*, because, in this area, relevant quantitative models are known precisely and the expert model may exist at the same time. Train operation is broadly classified into two control modes: (1) train speed regulation control and (2) train stopping control. For some systems, the conventional PID automatic train operation control hardwares were already being installed. The PID controller takes the error between the goal speed and the actual train speed to control the train's motor and brake. This requires a linerized system model, a desired state and an error criterion.

However, human can still control the train because they can evaluate the system objectives on the basis of experience. In the subway control system, temperature and pressure are state variables, decrease is a control action, and fuel is controlled. An expert (or operator) may provide a rule like "IF Temperature is High and Pressure is Slightly_high THEN decrease fuel." In this way, he (or she) evaluates the system state variables and decides on the control action which acts on the fuel. Moreover,

subjective performance indices such as riding comfort and accurate stopping can also be incorporated in the rule. A fuzzy controller based on such rules has already applied to the *Sendai* system in Japan in 1987.

Given these the expert rules and the PID models mentioned before, we may apply our methodology to check for consistency between these two models. When we find inconsistency, we may make slight alterations on the rules and (or) the membership functions in the fuzzy rules. Since most applications employing PID models usually can be replaced with experienced human operators and rules, this area is one of the most promising application areas to apply the presented methodology.

7.2.2 Application in MOOSE

For various application-oriented examples, we are particularly interested in the application within MOOSE (Multimodel Object Oriented Simulation Environment) [12, 11]. MOOSE is an enabling environment under development at University of Florida for modeling and simulation based on OOPM. OOPM extends object-oriented program design with visualization and reinforces the relation of “model” to “program.” This permits a tight coupling between a model author and the modeling and simulation process through an interactive HCI (Human Computer Interface). MOOSE consists of four major components [12]: **Modeler**, **Translator**, **Engine** and **Scenario**. **Modeler** interacts with a model author via a GUI in a way which helps the author make a valid conceptual model of the system. **Translator** is a bridge between a model design and a model execution. It reads the output from **Modeler** and automatically builds the corresponding structures of the conceptual model with C++ code, therefore it ensures that the program is a valid representation of the conceptual model. **Engine** is a C++ program, composed of **Translator** output plus runtime support, compiled and linked once, then repeatedly activated for the model execution. **Scenario** is a visualization-enabling GUI which interacts

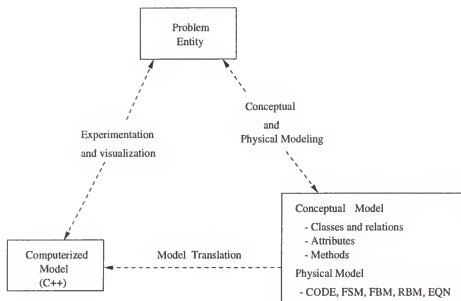


Figure 7.2. Modeling process in MOOSE

with **Engine** and displays **Engine**'s output in a meaningful way, so that the output of MOOSE can be validated against the author's expertise.

Even though this process reinforces the relation of the "model" to the "program" in a natural way, any adequate validation technique for the modeling process has not yet been developed. Therefore, by incorporating the method discussed in this research into MOOSE, we can obtain two types of benefits: one from validating the expert's rules against the simulation models, and another from validating the simulation models against the expert's knowledge. To make this point more understandable, we consider the modeling process in MOOSE as shown in Figure 7.2.

MOOSE supports many different types of models [12] including CODE, FSM (Finite State Machine), FBM (Functional Block Model), RBM (Rule Based Model) and EQN (EquationNal Constraint model) for the physical modeling process. Then, by translating the conceptual model into C++ code, it constructs the computerized model. MOOSE does not yet employ validation or verification techniques. Therefore, using the methodology introduced in this dissertation, the *face validation* process

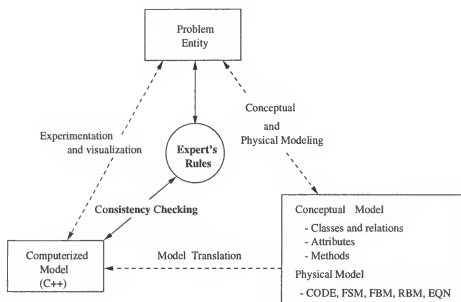


Figure 7.3. Consistency checking in MOOSE

with an expert can be automated, thereby contributing to validate the computerized models in MOOSE. A prerequisite for this process is that an expert's rule set for the system of interest must exist. Given that this condition is satisfied, the fuzzy simulation method can perform a *consistency check* between the expert's rules and the computerized model shown in Figure 7.3. Any inconsistency found can be considered to be due to an inadequate conceptual or physical model of MOOSE or an improperly translated physical model on the computer. To make this validation available, the following development steps are recommended:

1. make fuzzy simulation method available in MOOSE so that all quantitative models represented in *CODE*, *FSA*, *FBM*, *EQN* and *RBM* can be simulated using fuzzy sets.
2. develop a user interface in MOOSE for accepting the expert's fuzzy rules.
3. make a consistency-checking facility available between the expert rules and the computerized model by using the fuzzy simulation.

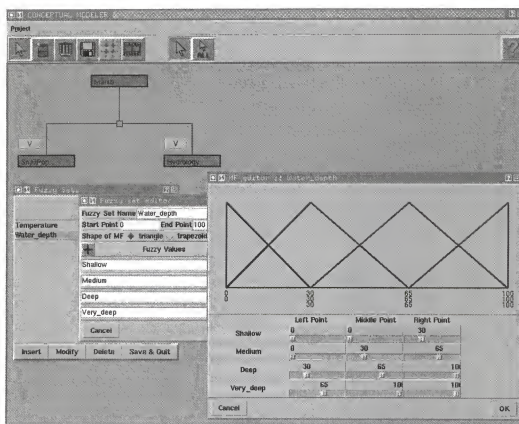


Figure 7.4. GUI for fuzzy simulation in MOOSE

4. develop a user interface via human intervention for resolving inconsistency.

Currently, fuzzy simulation is available in MOOSE, as shown in Figure 7.4. To proceed the remaining steps, especially step 3 and step 4, the approaches we discussed in Chapter 4 can be used. The consistency between two types of models (expert's rules and computerized models in MOOSE) can be measured by the difference between the CF presented by an expert and the CF calculated from the fuzzy simulation on each rule. This gives MOOSE's model author useful information, such as which components of the model should be further investigated. Consequently, by incorporating our method into MOOSE, we can obtain a benefit from validating the simulation models against the expert's knowledge.

CHAPTER 8 CONCLUSION

The motivation for this work lies with the problem of isolating the difference between qualitative and quantitative forms of models about physical systems. Comparing and contrasting the qualitative model, especially expert rules, with the quantitative model is viewed as being an integral part of an ongoing system validation procedure. The primary contribution of this work is that by presenting an interactive tool for checking for consistency and resolving inconsistency, we provide a methodology to serve as a *check and balance* to enhance the complex model validation process. For this purpose we introduced a fuzzy simulation method that bridges the gap between the two different levels of models and showed how we can directly compare and maintain these two models in a systematic manner using the fuzzy simulation method.

Since the uncertainty arising from the human reasoning process is easily represented by rules associated with confidence factors, we devised a way for replicating such processes by showing how the fuzzy simulation can derive confidence factors from quantitative models. To handle the **possibilistic uncertainty** arising from the human reasoning process, we've employed the *extension principle* in fuzzy set literature, various approximate reasoning tools such as the *rule of conjunctive or disjunctive composition* and the *weighted average method*.

Moreover, because the expert's qualitatively described rules naturally contain **linguistic vagueness**, we have devised our method to handle the expert's various levels of estimates, depending on the confidence level on his (or her) linguistic terms. When the expert's complete linguistic definitions are available *a priori*, a direct application

of fuzzy simulations gives us useful information, such as which knowledge components show inconsistency. Even without exact linguistic definitions, the method presented here determines consistency by searching for approximate fuzzy membership functions where two different levels of models maximally match. Whenever inconsistency is discovered in both cases, the possible sources for resolving inconsistency are suggested to humans by collecting information from fuzzy simulations or by conducting either an incremental test or a sensitivity test.

By devising a method of integrated qualitative and quantitative dynamical system knowledge refinement, we hope that this method serves as a *stepping stone* for developing more robust models, in general, about physical systems by exploiting knowledge at all levels, whether qualitative or quantitative.

REFERENCES

- [1] O. Balci. Validation, Verification, and Testing Techniques Throughout the Life Cycle of a Simulation Study. In J. D. Tew, S. Manivannan, D. A. Sadowski and A. F. Seila, editor, *WSC'94*, volume 8-11 December, pages 215-220, 1994.
- [2] H. R. Berenji. A Reinforcement Learning Based Architecture for Fuzzy Logic Control. *International J. of Approximate Reasoning*, 6(2):267-292, 1992.
- [3] H. R. Berenji and P. Khedkar. Learning and Tuning Fuzzy Logic Controllers Through Reinforcements. *IEEE Trans. on Neural Networks*, 3(5):724-740, 1992.
- [4] A. Bonarini and G. Bontempi. A Qualitative Simulation Approach for Fuzzy Dynamical Models. *acm Trans. on Modeling and Computer Simulation*, 4(4):285-313, 1994.
- [5] I. Bratko, I. Mozetič, and N. Lavrač. *KARDIO: A Study in Deep and Qualitative Knowledge for Expert Systems*. MIT Press, London, England, 1989.
- [6] L. A. Brita. A Knowledge-Based Approach for the Validation of Simulation Models: The Foundation. *ACM Trans. on Modeling and Computer Simulation*, 6(1):76-98, 1996.
- [7] L. G. Brita and F. N. Ozmizrak. A Knowledge-Based Approach for the Validation of Simulation Models: The Foundation. *ACM Trans. on Modeling and Computer Simulation*, 6(1):76-98, 1996.
- [8] F. E. Cellier. Qualitative Simulation of Technical Systems Using the General System Problem Solving Framework. *International J. of General Systems*, 13(4):333-344, 1987.
- [9] J. L. Chameau and J. C. Santamarina. Membership Functions I II. *International J. of Approximate Reasoning*, 1(3):287-301, 303-317, 1987.
- [10] E. Cox. Fuzzy Fundamentals. *IEEE Spectrum*, October:58-61, 1992.
- [11] R. M. Cubert and P. A. Fishwick. MOOSE: An Object-Oriented Multimodeling and Simulation Application Framework. volume June, 1997.
- [12] R. M. Cubert, T. Goktekin, and P. A. Fishwick. MOOSE: Architecture of an Object-Oriented Multimodeling Simulation System. In *SPIE AeroSense*, volume April, 1997.
- [13] V. Dhar and H. E. Pople. Rule-Based versus Structure-Based Models for Explaining and Generating Expert Behavior. *Communications of the ACM*, 30(6):542-555, 1987.

- [14] D. Dubois, H. Prade, and R.R. Yager. Basic Notions in Fuzzy Set Theory. In D. Dubois, H. Prade and R. R. Yager, editor, *Fuzzy Sets for Intelligent Systems*, pages 27-64. Morgan Kaufmann, 1993.
- [15] P. A. Fishwick. Fuzzy Set Methods for Qualitative and Natural Language Oriented Simulation. In O. Balci, R. P. Sadowski and R. E. Nance, editor, *WSC'90*, volume 9-12 December, pages 513-519, 1990.
- [16] P. A. Fishwick. Extracting Rules from Fuzzy Simulation. *Expert Systems With Applications*, 3:317-327, 1991.
- [17] P. A. Fishwick. Fuzzy Simulation: Specifying and Identifying Qualitative Models. *Int. J. General System*, 19:295-316, 1991.
- [18] P. A. Fishwick. *Simulation Model Design and Execution: Building Digital Worlds*. Prentice Hall, New Jersey, U.S.A., 1995.
- [19] R. Giles. The Concept of Grade Membership. In D. Dubois, H. Prade and R. R. Yager, editor, *Fuzzy Sets for Intelligent Systems*, pages 874-887. Morgan Kaufmann Publishers, Inc., 1993.
- [20] M. Ginsberg. *Essentials of Artificial Intelligence*. Morgan Kaufmann, San Mateo, California, 1993.
- [21] A. J. Gonzalez and D. D. Dankel. *The Engineering of Knowledge-based Systems: Theory and Practice*. Prentice Hall, New Jersey, U.S.A., 1993.
- [22] L. O. Hall and A. Kandel. The Evolution From Expert Systems to Fuzzy Expert Systems. In A. Kandel, editor, *Fuzzy Expert Systems*, pages 3-21. CRC Press, 1992.
- [23] H. Hellendoorn and D. Driankov. *Fuzzy Model Identification: Selected Approaches*. Springer, Berlin, Germany, 1997.
- [24] G. Kim and P. A. Fishwick. A Method for Resolving the Consistency Problem Between Rule-Based and Quantitative Model using Fuzzy Simulation. In *SPIE AeroSense '97 Conference.*, volume April 22-24, 1997.
- [25] G. Kim and P. A. Fishwick. A Validation Method using Fuzzy Simulation in an Object Oriented Physical Modeling Framework. In *SPIE AeroSense '98 Conference.*, volume April 14-16, 1998.
- [26] G. Kim and P. A. Fishwick. Enhancing Model Validation By Isolating Inconsistent Knowledge Between Fuzzy Rule-Based and Quantitative Models using Fuzzy Simulation. *submitted to IEEE Trans. Systems, Mans and Cybernetics*, 1998.
- [27] G. J. Klir and B. Yuan. *Fuzzy Sets and Fuzzy Logic: Theory and Applications*. Prentice Hall, New Jersey, U.S.A., 1995.
- [28] B. Kosko. *Fuzzy Engineering*. Prentice Hall, New Jersey, U.S.A., 1997.
- [29] A. M. Law and W. D. Kelton. *Simulation Modeling and Analysis*. McGraw-Hill, Inc., New York, 1991.

- [30] K. Lee and P. A. Fishwick. A Semi-automated Method for Dynamic Model Abstraction. In *SPIE AeroSense '97 Conference.*, volume April 22-24, 1997.
- [31] D. Li and F. E. Cellier. Fuzzy Measures in Inductive Reasoning. In O. Balci, R. P. Sadowski and R. E. Nance, editor, *WSC'90*, volume 9-12 December, pages 527-538, 1990.
- [32] P. Lindskog. Fuzzy Identification from a Grey Box Modeling Point of View. In H. Hellendoorn and D. Driankov, editor, *Fuzzy Model Identification*, pages 3-50. Springer, 1997.
- [33] A. M. Norwich and I. B. Turksen. A Model for the Measurement of Membership and the Consequences of its Empirical Implementation. In D. Dubois, H. Prade and R. R. Yager, editor, *Fuzzy Sets for Intelligent Systems*, pages 861-873. Morgan Kaufmann Publishers, Inc., 1993.
- [34] R. O'Keefe. Simulation and Expert Systems - A Taxonomy and Some Examples. *Simulation*, 46(1):10-16, 1986.
- [35] R. O'Keefe, O. Balci, and E. P. Smith. Validating expert system performance. *IEEE Expert*, Winter:81-89, 1987.
- [36] K. M. Passino and S. Yurkovich. *Fuzzy Control*. Addison-Wesley, Wokingham, England, 1998.
- [37] D. W. Patterson. *Introduction to Artificial Intelligence and Expert Systems*. Prentice Hall, Englewood Cliffs, New Jersey, 1990.
- [38] W. Pedrycz. An Identification Algorithm in Fuzzy Relational Systems. *Fuzzy Sets and Systems*, 13:153-167, 1984.
- [39] E. Rich and K. Knight. *Artificial Intelligence*, 2nd. ed. McGraw-Hill, New York, 1991.
- [40] S. J. Russell and P. Norvig. *Artificial Intelligence, a Modern Approach*. Prentice Hall, Englewood Cliffs, New Jersey, 1995.
- [41] T. L. Saaty. Measuring the Fuzziness of Sets. *J. Cybernetics*, 4:53-61, 1974.
- [42] T. L. Saaty. Scaling the Membership Function. *European J. of Operational Research*, 25(3):320-329, 1986.
- [43] A. P. Sage. *Systems Engineering*. A Wiley-Interscience Publication, New York, 1992.
- [44] R. G. Sargent. An Exploration of Possibilities for Expert Aids in Model Validation. In M. S. Elzas, T. I. Ören and B. P. Zeigler, editor, *Modeling and Simulation Methodology in the Artificial Intelligence Era*, pages 279-297. Elsevier Science, 1986.
- [45] R. G. Sargent. Simulation Model Verification and Validation. In B. L. Nelson, W. D. Kelton and G. M. Clark, editor, *WSC'91*, volume 8-11 December, pages 37-47, 1991.

- [46] Q. Shen and R. Leitch. Fuzzy Qualitative Simulation. *IEEE Trans. Systems, Mans and Cybernetics*, 23(4):1038-1061, 1993.
- [47] I. Sommerville. *Software Engineering*. Addison-Wesley, Wokingham, England, 1992.
- [48] L. Steels. Components of Expertise. *AI Magazine*, 11(2):28-49, 1990.
- [49] M. Sugeno and G. T. Kang. Structure Identification of Fuzzy Model. *Fuzzy Sets and Systems*, 28:15-33, 1988.
- [50] M. Sugeno and K. Tanaka. Successive Identification of a Fuzzy Model and Its Applications to Prediction of a Complex System. *Fuzzy Sets and Systems*, 42:315-334, 1991.
- [51] M. Sugeno and T. Yasukawa. A Fuzzy-Logic-Based Approach to Qualitative Modeling. *IEEE Trans. on Fuzzy Systems*, 1(1):7-31, 1993.
- [52] I. B. Turksen. Measurement of Membership Functions and Their Acquisition. *Fuzzy Sets and Systems*, 40(1):5-38, 1991.
- [53] P. J. Vesanterä and F. E. Cellier. Building Intelligence into an Autopilot-Using Qualitative Simulation to Support General Decision Making. *Simulations*, 52(3):111-121, 1989.
- [54] L. E. Widman and K. A. Loparo. Artificial Intelligence, Simulation, and Modeling: A Critical Survey. In L. E. Widman, K. A. Loparo and N. R. Nielsen, editor, *Artificial Intelligence, Simulation, and Modeling*, pages 1-44. A Wiley-Interscience, 1989.
- [55] W. F. Wolff. Microinteractive Predator-Prey Simulation. In W. F. Wolff, C. J. Soeder and F. R. Drepper, editor, *Proceedings of an International Workshop*, volume October, pages 285-306, 1987.
- [56] W. F. Wolff. An individual-Oriented Model of a Wading Bird Nesting Colony. *Ecological Modeling*, 72:75-114, 1994.
- [57] L. A. Zadeh. Fuzzy Sets. *Information and Control*, 8:328-353, 1965.
- [58] L. A. Zadeh. Outline of a New Approach to the Analysis of Complex Systems and Decision Processes. *IEEE Trans. Systems, Mans and Cybernetics*, SMC-3:28-44, 1973.
- [59] L. A. Zadeh. The Concept of a Linguistic Variable and Its Application to Approximate Reasoning I. *Information Sciences*, 8:199-251, 1975.
- [60] L. A. Zadeh. The Role of Fuzzy Logic in the Management of Uncertainty in Expert Systems. In R. R. Yager, editor, *Fuzzy Sets and Application: Selected Papers*, pages 413-442. Wiley, 1987.
- [61] H. J. Zimmermann. *Fuzzy Set Theory and Its Applications*. Kluwer-Nijhoff Publishing, MA, U.S.A., 1985.
- [62] R. Zwick. Combining Stochastic Uncertainty and Linguistic Inexactness: Theory and Experimental Evaluation of Four Fuzzy Probability Models. *Int. J. Man-Machine Studies*, 30(1):337-379, 1989.

BIOGRAPHICAL SKETCH

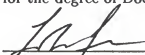
Gyooseok Kim received the B.S. degree in Electronics from Korean Air Force Academy in 1984 and the M.S. degree in Computer Science from Korean National Defense College in 1989. In earlier days, he served with Korean Air Force as a chief programmer of Korean Air Defense System. In 1998, he received a doctoral degree in the Computer and Information Science and Engineering department at the University of Florida. His major research area is modeling for computer simulation. His minor interests are fuzzy simulation, and knowledge acquisition and validation from qualitative and quantitative simulation.

I certify that I have read this study and that in my opinion it conforms to acceptable standards of scholarly presentation and is fully adequate, in scope and quality, as a dissertation for the degree of Doctor of Philosophy.



Paul A. Fishwick, Chairman
Associate Professor of Computer and
Information Science and Engineering

I certify that I have read this study and that in my opinion it conforms to acceptable standards of scholarly presentation and is fully adequate, in scope and quality, as a dissertation for the degree of Doctor of Philosophy.



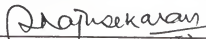
Li-Min Fu
Associate Professor of Computer and
Information Science and Engineering

I certify that I have read this study and that in my opinion it conforms to acceptable standards of scholarly presentation and is fully adequate, in scope and quality, as a dissertation for the degree of Doctor of Philosophy.



Douglas D. Dankel II
Assistant Professor of Computer and
Information Science and Engineering

I certify that I have read this study and that in my opinion it conforms to acceptable standards of scholarly presentation and is fully adequate, in scope and quality, as a dissertation for the degree of Doctor of Philosophy.



Sanguthevar Rajasekaran
Associate Professor of Computer and
Information Science and Engineering

I certify that I have read this study and that in my opinion it conforms to acceptable standards of scholarly presentation and is fully adequate, in scope and quality, as a dissertation for the degree of Doctor of Philosophy.



Sherman X. Bai
Assistant Professor of Industrial and
Systems Engineering

This dissertation was submitted to the Graduate Faculty of the College of Engineering and to the Graduate School and was accepted as partial fulfillment of the requirements for the degree of Doctor of Philosophy.

August 1998

Winfred M. Phillips
Dean, College of Engineering

Karen A. Holbrook
Dean, Graduate School